HOLTEK

2.4GHz Flash RF TX/RX MCU

BC66F840/BC66F850/BC66F860

Revision: V1.60 Date: November 26, 2019

www.holtek.com

Table of Contents

Features

CPU Features

- Operating voltage:
  - f_{SYS}=16MHz: 3.3V~5.5V
- 0.25μs instruction cycle with 16MHz system clock at V_{DD}=5V
- Power down and wake-up functions to reduce power consumption
- Oscillator types:
  - External 16MHz Crystal -- HXT
  - External 32.768kHz Crystal -- LXT
  - Internal 32kHz RC -- LIRC
- Multi-mode operation: NORMAL, SLOW, IDLE and SLEEP
- All instructions executed in one or two instruction cycles
- Table read instructions
- 63 powerful instructions
- Up to 12-level subroutine nesting
- Bit manipulation instruction

Peripheral Features

- Program Memory: Up to 16K×16
- RAM Data Memory: Up to 512×8
- True EEPROM Memory: Up to 256×8
- Watchdog Timer function
- Up to 35 bidirectional I/O lines
- Two pin-shared external interrupts
- Multiple Timer Module for time measure, input capture, compare match output, PWM output or single pulse output function
- Serial Interface Module – SIM (SPI/I^2C)
- Individual SPI interface – SPIA
- Comparator function
- Dual Time-Base functions for generation of fixed time interrupt signals
- Up to 16 channels 12-bit A/D Converter
- Flash program memory can be re-programmed up to 10,000 times
- Flash program memory data retention > 10 years
- True EEPROM data memory can be re-programmed up to 100,000 times
- True EEPROM data memory data retention > 10 years
- Low voltage reset function
- Low voltage detect function
- Package types: 32/40/46 pin QFN

RF Transceiver Features

- Low Power High Performance 2.4GHz GFSK Transceiver
- 2400-2483.5MHz ISM band operation
- Support 250Kbps, 1Mbps and 2Mbps air data rate
- Programmable output power
- Variable payload length from 1 to 32 bytes
- Automatic packet processing
- 6 data pipes for 1:6 star networks

General Description

The devices are Flash Memory type 8-bit high performance RISC architecture microcontrollers. Offering users the convenience of Flash Memory multi-programming features, these devices also include a wide range of functions and features. Other memory includes an area of RAM Data Memory as well as an area of true EEPROM memory for storage of non-volatile data such as serial numbers, calibration data etc.

Analog features include a multi-channel 12-bit A/D converter and a comparator functions. Multiple and extremely flexible Timer Modules provide timing, pulse generation and PWM generation functions. Protective features such as an internal Watchdog Timer, Low Voltage Reset and Low Voltage Detector coupled with excellent noise immunity and ESD protection ensure that reliable operation is maintained in hostile electrical environments.

A full choice of HXT, LXT and LIRC oscillator functions are provided including a fully integrated system oscillator which requires no external components for its implementation. The ability to operate and switch dynamically between a range of operating modes using different clock sources gives users the ability to optimise microcontroller operation and minimize power consumption.

The inclusion of flexible I/O programming features, Time-Base functions along with many other features ensure that the devices will find excellent use in applications such as electronic metering, environmental monitoring, handheld instruments, household appliances, electronically controlled tools, motor driving in addition to many others.

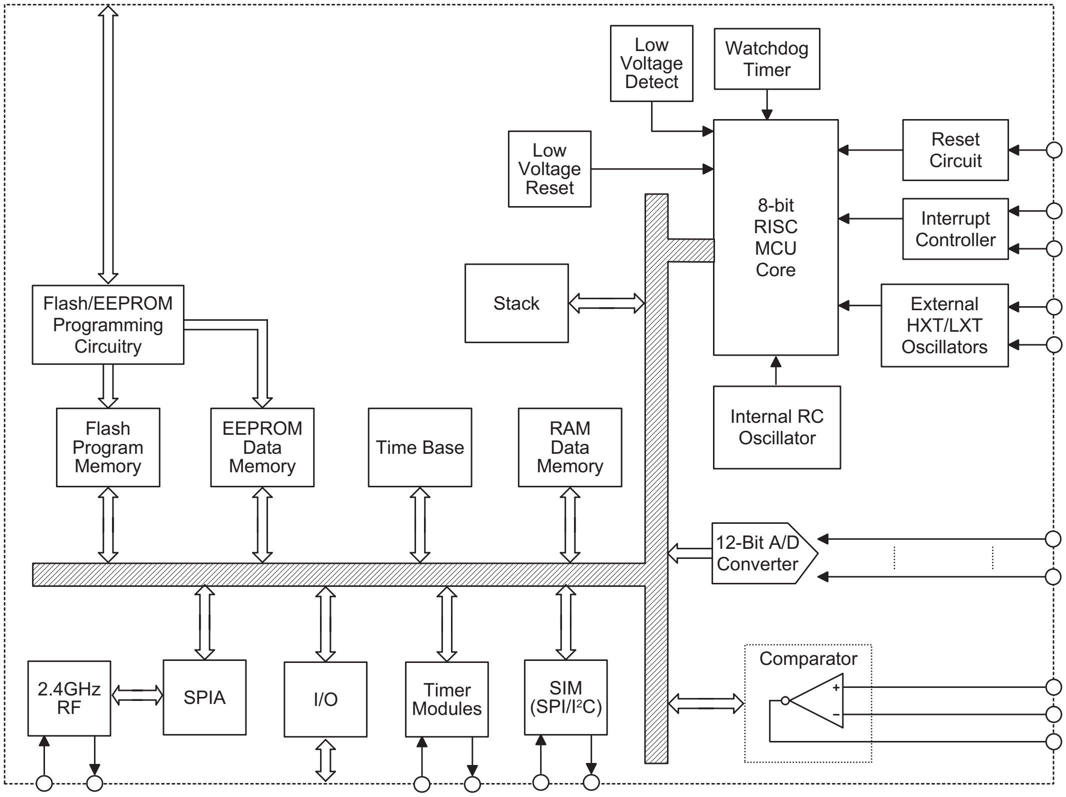
Selection Table

Most features are common to all devices, the main feature distinguishing them are Memory capacity, I/O count, A/D Converter channels, TM features, stack capacity and package types. The following table summarises the main features of each device.

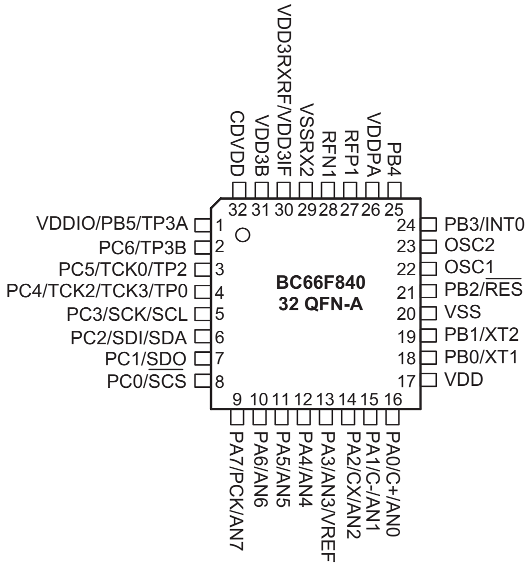
Part No.	Program Memory	Data Memory	EEPROM Memory	I/O	External Interrupt	A/D Converter
BC66F840	4k×16	256×8	128×8	21	2	12-bit×8
BC66F850	8k×16	384×8	256×8	29	2	12-bit×16
BC66F860	16k×16	512×8	256×8	35	2	12-bit×16

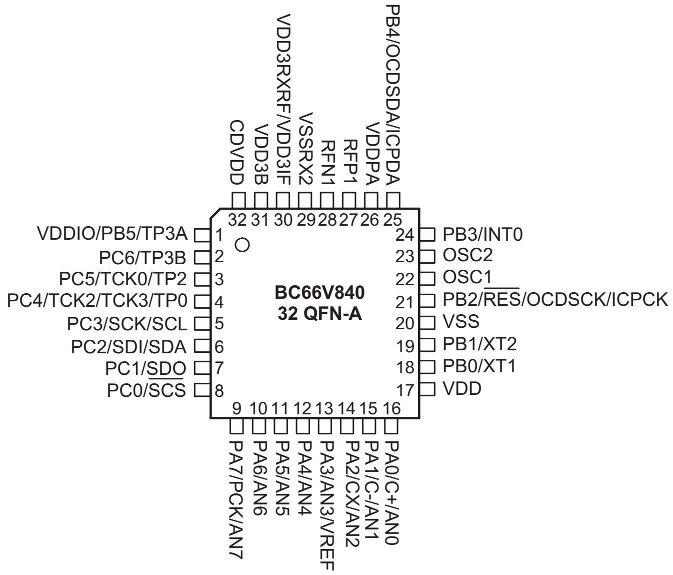
Part No.	Timer Module	SIM	SPIA	Time Base	Comparator	Stacks	Package
BC66F840	16-bit CTM×1 16-bit STM×1 16-bit ETM×1	√	√	2	1	8	32QFN
BC66F850	16-bit CTM×2 16-bit STM×1 16-bit ETM×1	√	√	2	1	8	40QFN
BC66F860	16-bit CTM×2 16-bit STM×1 16-bit ETM×1	√	√	2	1	12	46QFN

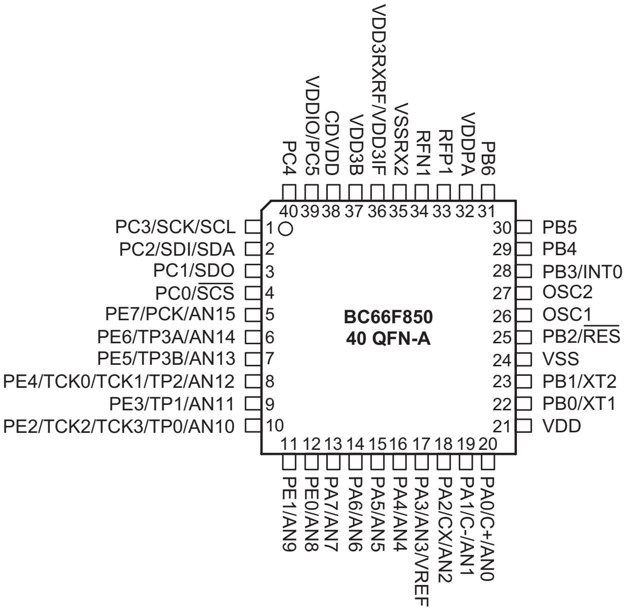
Block Diagram

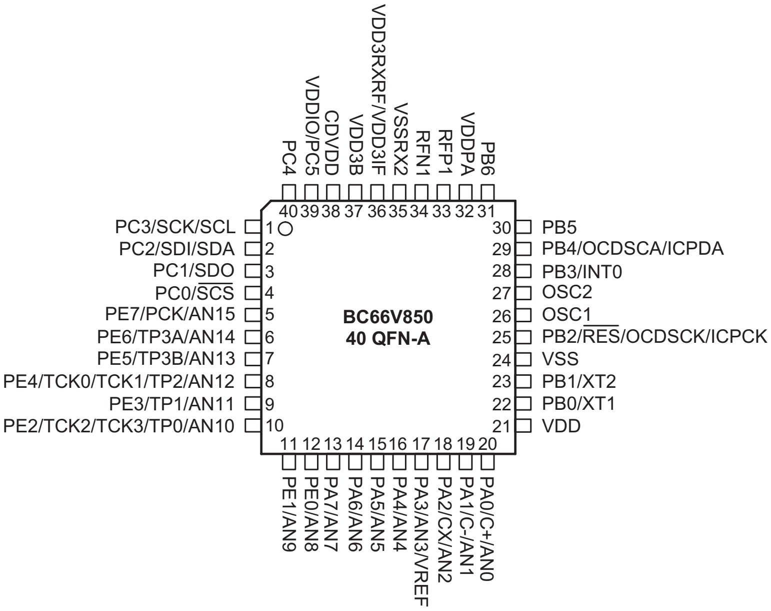


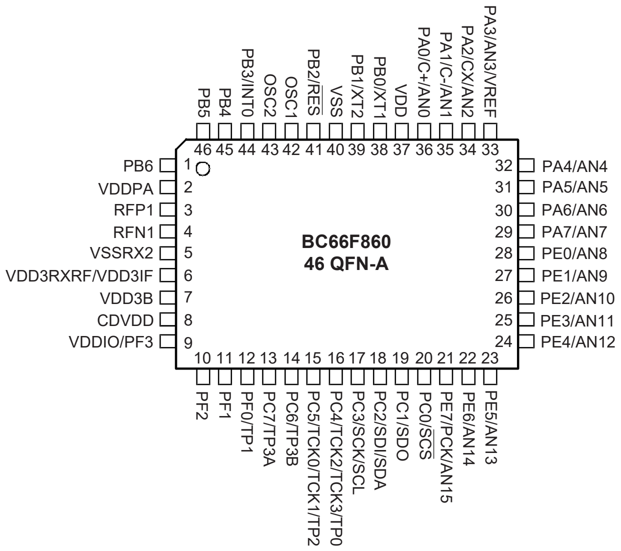
Pin Assignment

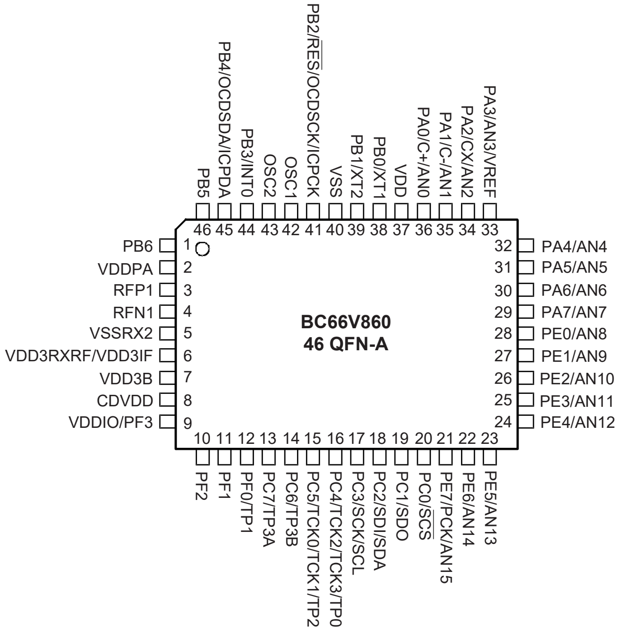





BC66F850
40 QFN-A
1 PC3/SCK/SCL
2 PC2/SDI/SDA
3 PC1/SDO
4 PC0/SCS
5 PE7/PCK/AN15
6 PE6/TP3A/AN14
7 PE5/TP3B/AN13
8 PE4/TCK0/TCK1/TP2/AN12
9 PE3/TP1/AN11
10 PE2/TCK2/TCK3/TP0/AN10
11 PE1/AN9
12 PE0/AN8
13 PA7/AN7
14 PA6/AN6
15 PA5/AN5
16 PA4/AN4
17 PA3/AN3/VREF
18 PA2/CX/AN2
19 PA1/C-/AN1
20 PA0/C+/AN0
21 VDD
22 PB0/XT1
23 PB1/XT2
24 VSS
25 PB2/RES
26 OSC1
27 OSC2
28 PB3/INT0
29 PB4
30 PB5
31 PB6
32 VDDPA
33 RFP1
34 RFN1
35 VSSRX2
36 VDD3RXRF/VDD3IF
37 VDD3B
38 CDVDD
39 VDDIO/PC5
40 PC4


BC66V850
40 QFN-A
1 PC3/SCK/SCL
2 PC2/SDI/SDA
3 PC1/SDO
4 PC0/SCS
5 PE7/PCK/AN15
6 PE6/TP3A/AN14
7 PE5/TP3B/AN13
8 PE4/TCK0/TCK1/TP2/AN12
9 PE3/TP1/AN11
10 PE2/TCK2/TCK3/TP0/AN10
11 PE1/AN9
12 PE0/AN8
13 PA7/AN7
14 PA6/AN6
15 PA5/AN5
16 PA4/AN4
17 PA3/AN3/VREF
18 PA2/CX/AN2
19 PA1/C-/AN1
20 PA0/C+/AN0
21 VDD
22 PB0/XT1
23 PB1/XT2
24 VSS
25 PB2/RES/OCDSCK/ICPCK
26 OSC1
27 OSC2
28 PB3/INT0
29 PB4/OCDSCA/ICPDA
30 PB5
31 PB6
32 VDDPA
33 RFP1
34 RFN1
35 VSSRX2
36 VDD3RXRF/VDD3IF
37 VDD3B
38 CDVDD
39 VDDIO/PC5
40 PC4


BC66F860
46 QFN-A
1 PB6
2 VDDPA
3 RFP1
4 RFN1
5 VSSRX2
6 VDD3RXRF/VDD3IF
7 VDD3B
8 CDVDD
9 VDDIO/PF3
10 PF2
11 PF1
12 PF0/TP1
13 PC7/TP3A
14 PC6/TP3B
15 PC5/TCK0/TCK1/TP2
16 PC4/TCK2/TCK3/TP0
17 PC3/SCK/SCL
18 PC2/SDI/SDA
19 PC1/SDO
20 PC0/SCS
21 PE7/PCK/AN15
22 PE6/AN14
23 PE5/AN13
24 PE4/AN12
25 PE3/AN11
26 PE2/AN10
27 PE1/AN9
28 PE0/AN8
29 PA7/AN7
30 PA6/AN6
31 PA5/AN5
32 PA4/AN4
33 PA3/AN3/VREF
34 PA2/CX/AN2
35 PA1/C-/AN1
36 PA0/C+/AN0
37 VDD
38 PB0/XT1
39 PB1/XT2
40 VSS
41 PB2/RES
42 OSC1
43 OSC2
44 PB3/INT0
45 PB4
46 PB5


BC66V860
46 QFN-A
1 PB6
2 VDDPA
3 RFP1
4 RFN1
5 VSSRX2
6 VDD3RXRF/VDD3IF
7 VDD3B
8 CDVDD
9 VDDIO/PF3
10 PF2
11 PF1
12 PF0/TP1
13 PC7/TP3A
14 PC6/TP3B
15 PC5/TCK0/TCK1/TP2
16 PC4/TCK2/TCK3/TP0
17 PC3/SCK/SCL
18 PC2/SDI/SDA
19 PC1/SDO
20 PC0/SCS
21 PE7/PCK/AN15
22 PE6/AN14
23 PE5/AN13
24 PE4/AN12
25 PE3/AN11
26 PE2/AN10
27 PE1/AN9
28 PE0/AN8
29 PA7/AN7
30 PA6/AN6
31 PA5/AN5
32 PA4/AN4
33 PA3/AN3/VREF
34 PA2/CX/AN2
35 PA1/C-/AN1
36 PA0/C+/AN0
37 VDD
38 PB0/XT1
39 PB1/XT2
40 VSS
41 PB2/RES/OCDSCK/ICPCK
42 OSC1
43 OSC2
44 PB3/INT0
45 PB4/OCDSDA/ICPDA
46 PB5

Pin Descriptions

With the exception of the power pins, all pins on these devices can be referenced by their Port name, e.g. PA0, PA1, etc, which refer to the digital I/O function of the pins. However some of these Port pins are also shared with other function such as the Analog to Digital Converter, Timer Module pins etc. The function of each pin is listed in the following table, however the details behind how each pin is configured is contained in other sections of the datasheet.

The BC66V8x0 device is the corresponding EV chip of the BC66F8x0 device. It supports the "On-Chip Debug" function for debugging during development using the OCDSDA and OCDSCK pins connected to the Holtek HT-IDE development tools.

BC66F840/BC66V840

Pin Name	Function	OP	I/T	O/T	Pin-Shared Mapping
Microcontroller Pins					
PA0~PA7	Port A	PAWU PAPU	ST	CMOS	—
PB0~PB5	Port B	PBPU	ST	CMOS	—
PC0~PC6	Port C	PCPU	ST	CMOS	—
PD0~PD5	Port D, internally connected to RF transceiver	PDPU	ST	CMOS	—
CLKO	Clock output, internally connected to RF transceiver	CTRL1	—	CMOS	—
AN0~AN7	A/D Converter input	ACERL	AN	—	PA0~PA7
VREF	A/D Converter reference input	ADCR1	AN	—	PA3
C+, C-	Comparator input	CPC	AN	—	PA0, PA1
CX	Comparator output	CPC	—	CMOS	PA2
TCK0	TM0 input	—	ST	—	PC5
TCK2~TCK3	TM2, TM3 input	—	ST	—	PC4
TP0, TP2	TM0, TM2 I/O	TMPC	ST	CMOS	PC4, PC5
TP3A, TP3B	TM3 I/O	TMPC	ST	CMOS	PB5, PC6
INT0	External Interrupt 0	INTEG INTC0	ST	—	PB3
INT1	External Interrupt 1 Internally connected to RF transceiver	INTEG INTC0	ST	—	PD2
PCK	Peripheral Clock output	—	—	CMOS	PA7
SDI	SPI Data input	—	ST	—	PC2
SDO	SPI Data output	—	—	CMOS	PC1
$\overline{\text{SCS}}$	SPI Slave Select	—	ST	CMOS	PC0
SCK	SPI Serial Clock	—	ST	CMOS	PC3
SCL	I^{2}C Clock	—	ST	NMOS	PC3
SDA	I^{2}C Data	—	ST	NMOS	PC2
OSC1	HXT pin	CO	HXT	—	—
OSC2	HXT pin	CO	—	HXT	—
XT1	LXT pin	CO	LXT	—	PB0
XT2	LXT pin	CO	—	LXT	PB1
$\overline{\text{RES}}$	Reset pin	CO	ST	—	PB2
VDDIO	Power supply for interconnected pins	—	PWR	—	PB5
VDD	Positive Power supply	—	PWR	—	—

Pin Name	Function	OP	I/T	O/T	Pin-Shared Mapping
VSS	Negative power supply, ground	—	PWR	—	—
OCDSCK	OCDS clock pin, for EV chip used only	—	—	CMOS	PB2
OCDSDA	OCDS data pin, for EV chip used only	—	ST	NMOS	PB4
ICPCK	ICP clock pin, for EV chip used only	—	—	CMOS	PB2
ICPDA	ICP data pin, for EV chip used only	—	ST	NMOS	PB4
2.4GHz RF Transceiver Pins					
RFP1	RF positive input/output port	—	—	—	—
RFN1	RF negative input/output port	—	—	—	—
VDDPA	1.8V Regulator output for Power Amplifier	—	—	—	—
VDD3RXRF/ VDD3IF	RF Transceiver positive power supply	—	PWR	—	—
VDD3B	RF Transceiver positive power supply	—	PWR	—	—
CDVDD	1.8V regulator output decoupling capacitor pin	—	—	—	—
VSSRX2	RF Transceiver negative power supply, ground	—	PWR	—	—

Note: I/T: Input type; O/T: Output type
OP: Optional by configuration option (CO) or register option
PWR: Power; CO: Configuration option; ST: Schmitt Trigger input
CMOS: CMOS output; NMOS: NMOS output
SCOM: Software controlled LCD COM; AN: Analog input pin
HXT: High frequency crystal oscillator
LXT: Low frequency crystal oscillator

BC66F850/BC66V850

Pin Name	Function	OP	I/T	O/T	Pin-Shared Mapping
Microcontroller Pins					
PA0~PA7	Port A	PAWU PAPU	ST	CMOS	—
PB0~PB6	Port B	PBPU	ST	CMOS	—
PC0~PC5	Port C	PCPU	ST	CMOS	—
PD0~PD5	Port D, internally connected to RF transceiver	PDPU	ST	CMOS	—
CLKO	Clock output, internally connected to RF transceiver	CTRL1	—	CMOS	—
PE0~PE7	Port E	PEPU	ST	CMOS	—
AN0~AN15	A/D Converter input	ACERL	AN	—	PA0~PA7, PE0~PE7
VREF	A/D Converter reference input	ADCR1	AN	—	PA3
C+, C-	Comparator input	CPC	AN	—	PA0, PA1
CX	Comparator output	CPC	—	CMOS	PA2
TCK0~TCK1	TM0, TM1 input	—	ST	—	PE4
TCK2~TCK3	TM2, TM3 input	—	ST	—	PE2
TP0~TP2	TM0~TM2 I/O	TMPC	ST	CMOS	PE2~PE4
TP3A, TP3B	TM3 I/O	TMPC	ST	CMOS	PE6, PE5

Pin Name	Function	OP	I/T	O/T	Pin-Shared Mapping
INT0	External Interrupt 0	INTEG INTC0	ST	—	PB3
INT1	External Interrupt 1 Internally connected to RF transceiver	INTEG INTC0	ST	—	PD2
PCK	Peripheral Clock output	—	—	CMOS	PE7
SDI	SPI Data input	—	ST	—	PC2
SDO	SPI Data output	—	—	CMOS	PC1
$\overline{\text{SCS}}$	SPI Slave Select	—	ST	CMOS	PC0
SCK	SPI Serial Clock	—	ST	CMOS	PC3
SCL	I^{2}C Clock	—	ST	NMOS	PC3
SDA	I^{2}C Data	—	ST	NMOS	PC2
OSC1	HXT pin	CO	HXT	—	—
OSC2	HXT pin	CO	—	HXT	—
XT1	LXT pin	CO	LXT	—	PB0
XT2	LXT pin	CO	—	LXT	PB1
$\overline{\text{RES}}$	Reset pin	CO	ST	—	PB2
VDDIO	Power supply for interconnected pins	—	PWR	—	PC5
VDD	Positive Power supply	—	PWR	—	—
VSS	Negative power supply, ground	—	PWR	—	—
OCDSCK	OCDS clock pin, for EV chip used only	—	—	CMOS	PB2
OCDSDA	OCDS data pin, for EV chip used only	—	ST	NMOS	PB4
ICPCK	ICP clock pin, for EV chip used only	—	—	CMOS	PB2
ICPDA	ICP data pin, for EV chip used only	—	ST	NMOS	PB4
2.4GHz RF Transceiver Pins					
RFP1	RF positive input/output port	—	—	—	—
RFN1	RF negative input/output port	—	—	—	—
VDDPA	1.8V Regulator output for Power Amplifier	—	—	—	—
VDD3RXRF/ VDD3IF	RF Transceiver positive power supply	—	PWR	—	—
VDD3B	RF Transceiver positive power supply	—	PWR	—	—
CDVDD	1.8V regulator output decoupling capacitor pin	—	—	—	—
VSSRX2	RF Transceiver negative power supply, ground	—	PWR	—	—

Note: I/T: Input type; O/T: Output type
OP: Optional by configuration option (CO) or register option
PWR: Power; CO: Configuration option; ST: Schmitt Trigger input
CMOS: CMOS output; NMOS: NMOS output
SCOM: Software controlled LCD COM; AN: Analog input pin
HXT: High frequency crystal oscillator
LXT: Low frequency crystal oscillator

BC66F860/BC66V860

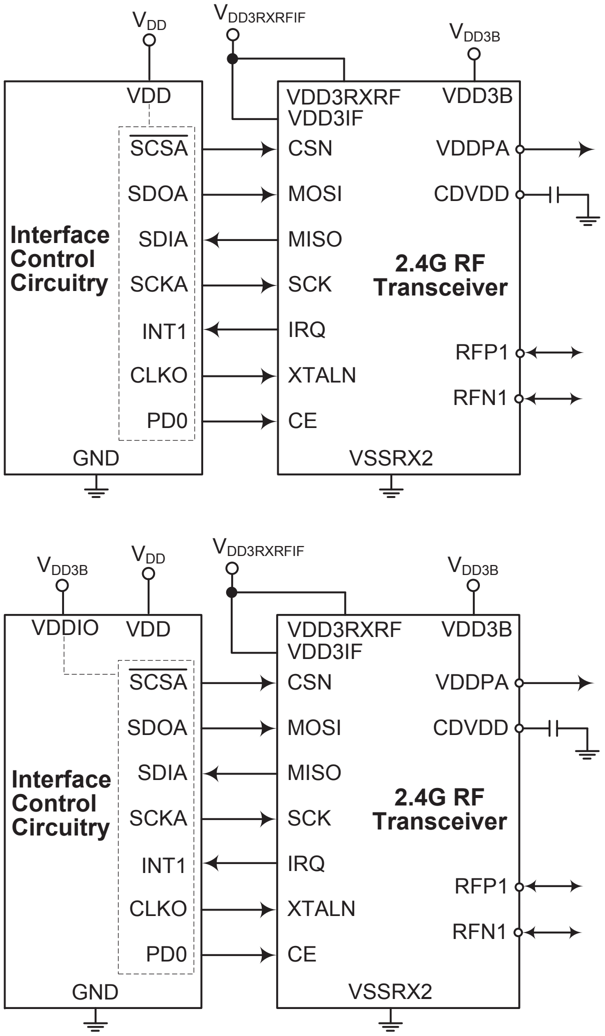
Pin Name	Function	OP	I/T	O/T	Pin-Shared Mapping
Microcontroller Pins					
PA0~PA7	Port A	PAWU PAPU	ST	CMOS	—
PB0~PB6	Port B	PBPU	ST	CMOS	—
PC0~PC5	Port C	PCPU	ST	CMOS	—
PD0~PD5	Port D, internally connected to RF transceiver	PDPU	ST	CMOS	—
CLKO	Clock output, internally connected to RF transceiver	CTRL1	—	CMOS	—
PE0~PE7	Port E	PEPU	ST	CMOS	—
AN0~AN15	A/D Converter input	ACERL	AN	—	PA0~PA7, PE0~PE7
VREF	A/D Converter reference input	ADCR1	AN	—	PA3
C+, C-	Comparator input	CPC	AN	—	PA0, PA1
CX	Comparator output	CPC	—	CMOS	PA2
TCK0~TCK1	TM0, TM1 input	—	ST	—	PC5
TCK2~TCK3	TM2, TM3 input	—	ST	—	PC4
TP0~TP2	TM0~TM2 I/O	TMPC	ST	CMOS	PC4, PF0, PC5
TP3A, TP3B	TM3 I/O	TMPC	ST	CMOS	PC6, PC7
INT0	External Interrupt 0	INTEG INTC0	ST	—	PB3
INT1	External Interrupt 1 Internally connected to RF transceiver	INTEG INTC0	ST	—	PD2
PCK	Peripheral Clock output	—	—	CMOS	PE7
SDI	SPI Data input	—	ST	—	PC2
SDO	SPI Data output	—	—	CMOS	PC1
$\overline{\text{SCS}}$	SPI Slave Select	—	ST	CMOS	PC0
SCK	SPI Serial Clock	—	ST	CMOS	PC3
SCL	I^{2}C Clock	—	ST	NMOS	PC3
SDA	I^{2}C Data	—	ST	NMOS	PC2
OSC1	HXT pin	CO	HXT	—	—
OSC2	HXT pin	CO	—	HXT	—
XT1	LXT pin	CO	LXT	—	PB0
XT2	LXT pin	CO	—	LXT	PB1
$\overline{\text{RES}}$	Reset pin	CO	ST	—	PB2
VDDIO	Power supply for interconnected pins	—	PWR	—	PF3
VDD	Positive Power supply	—	PWR	—	—
VSS	Negative power supply, ground	—	PWR	—	—
OCDSCK	OCDS clock pin, for EV chip used only	—	—	CMOS	PB2
OCDSDA	OCDS data pin, for EV chip used only	—	ST	NMOS	PB4
ICPCK	ICP clock pin, for EV chip used only	—	—	CMOS	PB2
ICPDA	ICP data pin, for EV chip used only	—	ST	NMOS	PB4

Pin Name	Function	OP	I/T	O/T	Pin-Shared Mapping
2.4GHz RF Transceiver Pins					
RFP1	RF positive input/output port	—	—	—	—
RFN1	RF negative input/output port	—	—	—	—
VDDPA	1.8V Regulator output for Power Amplifier	—	—	—	—
VDD3RXRF/ VDD3IF	RF Transceiver positive power supply	—	PWR	—	—
VDD3B	RF Transceiver positive power supply	—	PWR	—	—
CDVDD	1.8V regulator output decoupling capacitor pin	—	—	—	—
VSSRX2	RF Transceiver negative power supply, ground	—	PWR	—	—

Note: I/T: Input type; O/T: Output type
OP: Optional by configuration option (CO) or register option
PWR: Power; CO: Configuration option; ST: Schmitt Trigger input
CMOS: CMOS output; NMOS: NMOS output
SCOM: Software controlled LCD COM; AN: Analog input pin
HXT: High frequency crystal oscillator
LXT: Low frequency crystal oscillator

2.4GHz RF Transceiver IP Internal Control Signals

Internal Control Signal		2.4GHz RF Transceiver IP	
Signal Name	Signal Type	Signal Name	Signal Type
PD0	O	CE	I
PD1/$\overline{\text{SCSA}}$	O	CSN	I
PD2/INT1	I	IRQ	O
PD3/SDIA	I	MISO	O
PD4/SDOA	O	MOSI	I
PD5/SCKA	O	SCK	I
CLKO	O	XTALN	I



2.4GHz RF Transceiver IP Control Diagram

Absolute Maximum Ratings

Supply Voltage	V_{SS}-0.3V to V_{SS}+6.0V
Storage Temperature	-50˚C to 150˚C
Input Voltage	V_{SS}-0.3V to V_{DD}+0.3V
Operating Temperature	-40˚C to 85˚C
I_{OL} Total	100mA
I_{OH} Total	-100mA
Total Power Dissipation	500mW
ESD HBM	±1.5KV
ESD MM	±100V

Note: These are stress ratings only. Stresses exceeding the range specified under "Absolute Maximum Ratings" may cause substantial damage to the device. Functional operation of this device at other conditions beyond those listed in the specification is not implied and prolonged exposure to extreme conditions may affect device reliability.

*Device is ESD sensitive. HBM (Human Body Mode) is based on MIL-STD-883H Method 3015.8. MM (Machine Mode) is based on JEDEC EIA/JESD22-A115.

D.C. Characteristics

Ta=25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD1}	Operating Voltage (BC66F840) (HXT=16MHz)	—	f_{SYS}=4MHz	2.2	—	5.5	V
			f_{SYS}=8MHz	2.4	—	5.5	V
			f_{SYS}=16MHz	3.0	—	5.5	V
V_{DD2}	Operating Voltage (BC66F850) (HXT=16MHz)	—	f_{SYS}=4MHz	2.2	—	5.5	V
			f_{SYS}=8MHz	2.4	—	5.5	V
			f_{SYS}=16MHz	3.3	—	5.5	V
V_{DD3}	Operating Voltage (BC66F860) (HXT=16MHz)	—	f_{SYS}=4MHz	2.2	—	5.5	V
			f_{SYS}=8MHz	2.4	—	5.5	V
			f_{SYS}=16MHz	3.6	—	5.5	V
V_{DDIO}	Operating Voltage (For Port D & CLKO Internal Signals)	—	—	2.2	—	5.5	V
I_{DD1}	Operating Current, Normal Mode, $f_{SYS}=f_H=f_{HXT}$	3V	No load, f_H=16MHz, ADC off, WDT enable	—	2.0	3.0	mA
		5V		—	4.5	7.0	mA
I_{DD2}	Operating Current, Slow Mode, $f_{SYS}=f_L=f_{SUB}=f_{LXT}$	3V	No load, $f_{SYS}=f_{LXT}$, ADC off, WDT enable, LXTLP=0	—	40	60	μA
		5V		—	50	80	μA
		3V	No load, $f_{SYS}=f_{LXT}$, ADC off, WDT enable, LXTLP=1	—	40	60	μA
		5V		—	50	80	μA
I_{DD3}	Operating Current, Slow Mode, $f_{SYS}=f_L=f_{SUB}=f_{LIRC}$	3V	No load, $f_{SYS}=f_{LIRC}$, ADC off, WDT enable	—	10	20	μA
		5V		—	30	50	μA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{DD4}	Operating Current, Normal Mode, f_H=f_{HXT}=16MHz	3V	No load, f_{SYS}=f_H/2, ADC off, WDT enable	—	0.9	1.5	mA
		5V		—	2.5	3.75	mA
		3V	No load, f_{SYS}=f_H/4, ADC off, WDT enable	—	0.7	1.0	mA
		5V		—	2.0	3.0	mA
		3V	No load, f_{SYS}=f_H/8, ADC off, WDT enable	—	0.6	0.9	mA
		5V		—	1.6	2.4	mA
		3V	No load, f_{SYS}=f_H/16, ADC off, WDT enable	—	0.5	0.75	mA
		5V		—	1.5	2.25	mA
		3V	No load, f_{SYS}=f_H/32, ADC off, WDT enable	—	0.49	0.74	mA
		5V		—	1.45	2.18	mA
		3V	No load, f_{SYS}=f_H/64, ADC off, WDT enable	—	0.47	0.71	mA
		5V		—	1.4	2.1	mA
I_{IDLE1}	IDLE0 Mode Standby Current (LXT on)	3V	No load, ADC off, WDT enable, LXTLP=0	—	5	10	μA
		5V		—	16	32	μA
		3V	No load, ADC off, WDT enable, LXTLP=1	—	5	10	μA
		5V		—	16	32	μA
I_{IDLE2}	IDLE0 Mode Standby Current (LIRC on)	3V	No load, ADC off, WDT enable, LVR disable	—	1.3	3.0	μA
		5V		—	2.2	5.0	μA
I_{IDLE3}	IDLE1 Mode Standby Current (HXT)	3V	No load, ADC off, WDT enable, f_{SYS}=16MHz on	—	1.0	2.0	mA
		5V		—	2.0	4.0	mA
I_{SLEEP1}	SLEEP0 Mode Standby Current (LIRC off)	3V	No load, ADC off, WDT disable, LVR disable	—	0.1	1.0	μA
		5V		—	0.3	2.0	μA
I_{SLEEP2}	SLEEP1 Mode Standby Current (LXT on)	3V	No load, ADC off, WDT enable, LXTLP=0, LVR disable	—	5	10	μA
		5V		—	16	32	μA
I_{SLEEP3}	SLEEP1 Mode Standby Current (LXT on)	3V	No load, ADC off, WDT enable, LXTLP=1, LVR disable	—	5	10	μA
		5V		—	15	30	μA
I_{SLEEP4}	SLEEP1 Mode Standby Current (LIRC on)	3V	No load, ADC off, WDT enable, LVR disable	—	1.3	5.0	μA
		5V		—	2.2	10	μA
V_{IL1}	Input Low Voltage for I/O Ports or Input Pins except $\overline{RES}$ pin	5V	—	0	—	1.5	V
		—	—	0	—	$0.2V_{DD}$	V
V_{IH1}	Input High Voltage for I/O Ports or Input Pins except $\overline{RES}$ pin	5V	—	3.5	—	5.0	V
		—	—	$0.8V_{DD}$	—	V_{DD}	V
V_{IL2}	Input Low Voltage for $\overline{RES}$ pin	—	—	0	—	$0.4V_{DD}$	V
V_{IH2}	Input High Voltage for $\overline{RES}$ pin	—	—	$0.9V_{DD}$	—	V_{DD}	V
I_{OL1}	I/O Port Sink Current (Except Port D)	3V	V_{OL}=$0.1V_{DD}$	4	8	—	mA
		5V	V_{OL}=$0.1V_{DD}$	10	20	—	mA
I_{OH1}	I/O Port Source Current (Except Port D)	3V	V_{OH}=$0.9V_{DD}$	-2	-4	—	mA
		5V	V_{OH}=$0.9V_{DD}$	-5	-10	—	mA
I_{OL2}	I/O Port Sink Current (For Port D)	3V	V_{OL}=$0.1V_{DD}$ or $0.1V_{DDIO}$	2	4	—	mA
		5V	V_{OL}=$0.1V_{DD}$ or $0.1V_{DDIO}$	5	10	—	mA
I_{OH2}	I/O Port Source Current (For Port D)	3V	V_{OH}=$0.9V_{DD}$ or $0.9V_{DDIO}$	-1	-2	—	mA
		5V	V_{OH}=$0.9V_{DD}$ or $0.9V_{DDIO}$	-2.5	-5	—	mA
I_{OL3}	Sink Current for CLKO output	3V	V_{OL_CLKO}=$0.1V_{DD}$ or $0.1V_{DDIO}$	4.5	7	—	mA
		5V	V_{OL_CLKO}=$0.1V_{DD}$ or $0.1V_{DDIO}$	10.5	14	—	mA
I_{OH3}	Source Current for CLKO output	3V	V_{OH_CLKO}=$0.9V_{DD}$ or $0.9V_{DDIO}$	-3	-4	—	mA
		5V	V_{OH_CLKO}=$0.9V_{DD}$ or $0.9V_{DDIO}$	-6	-8	—	mA
R_{PH}	Pull-high Resistance for I/O Ports	3V	—	20	60	100	kΩ
		5V	—	10	30	50	kΩ

A.C. Characteristics

Ta=25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS}	Operating clock (HXT)	3.3V~5.5V	—	—	16	—	MHz
f_{LXT}	32768Hz Crystal Oscillator Clock	—	—	—	32768	—	kHz
f_{LIRC}	Low Speed Internal RC Oscillator Clock	5V	Ta=25˚C	-10%	32	+10%	kHz
		2.2V~5.5V	Ta=-40˚C to 85˚C	-30%	32	+60%	kHz
t_{TCK}	TCKn Input Pulse Width	—	—	0.3	—	—	μs
t_{RES}	External Reset Low Pulse Width	—	—	10	—	—	μs
t_{INT}	Interrupt Pulse Width	—	—	10	—	—	μs
t_{EERD}	EEPROM Read Time	—	—	—	2	4	t_{SYS}
t_{EEWR}	EEPROM Write Time	—	—	—	2	4	ms
t_{SST}	System Start-up Timer Period (Wake-up from HALT, f_{SYS} on at HALT state)	—	—	2	—	—	t_{SYS}
	System Start-up Timer Period (Wake-up from HALT, f_{SYS} off at HALT state)	—	f_{SYS}=HXT	128	—	—	t_{SYS}
		—	f_{SYS}=LIRC	2	—	—	t_{SYS}
t_{RSTD}	System Reset Delay Time (Power On Reset)	—	—	25	50	100	ms
	System Reset Delay Time (Any Reset except Power On Reset)	—	—	8.3	16.7	33.3	ms

Note: t_{SYS}=1/f_{SYS}

LVD & LVR Electrical Characteristics

Ta=25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{LVR1}	Low Voltage Reset Voltage	—	LVR Enable, 2.1V option	-5%× Typ.	2.1	+5%× Typ.	V
V_{LVR2}			LVR Enable, 2.55V option		2.55		V
V_{LVR3}			LVR Enable, 3.15V option		3.15		V
V_{LVR4}			LVR Enable, 3.8V option		3.8		V
V_{LVD1}	Low Voltage Detector Voltage	—	LVDEN=1, V_{LVD}=2.0V	-5%× Typ.	2.0	+5%× Typ.	V
V_{LVD2}			LVDEN=1, V_{LVD}=2.2V		2.2		V
V_{LVD3}			LVDEN=1, V_{LVD}=2.4V		2.4		V
V_{LVD4}			LVDEN=1, V_{LVD}=2.7V		2.7		V
V_{LVD5}			LVDEN=1, V_{LVD}=3.0V		3.0		V
V_{LVD6}			LVDEN=1, V_{LVD}=3.3V		3.3		V
V_{LVD7}			LVDEN=1, V_{LVD}=3.6V		3.6		V
V_{LVD8}			LVDEN=1, V_{LVD}=4.0V		4.0		V
I_{LVR}	Additional Power Consumption if LVR is used	3V	LVR disable→LVR enable	—	30	45	μA
		5V		—	60	90	μA
I_{LVD}	Additional Power Consumption if LVD is used	3V	LVD disable→LVD enable (LVR disable)	—	40	60	μA
		5V		—	75	115	μA
t_{LVR}	Low Voltage Width to Reset	—	—	120	240	480	μs
t_{LVD}	Low Voltage Width to Interrupt	—	—	20	45	90	μs
t_{LVDS}	LVDO stable time	—	For LVR enable, LVD off→on	15	—	—	μs
		—	For LVR disable, LVD off→on	15	—	—	μs
t_{SRESET}	Software Reset Width to Reset	—	—	45	90	120	μs

ADC Characteristics

Ta=25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{ADI}	A/D Converter Input Voltage	—	—	0	—	V_{REF}	V
V_{REF}	A/D Converter Reference Voltage	—	—	2	—	V_{DD}	V
V_{BG}	Reference Voltage with Buffer Voltage	—		-3%	1.09	+3%	V
DNL1	Differential Non-linearity	—	V_{REF}=V_{DD}, t_{ADCK}=0.5μs Ta=25˚C	-3	—	+3	LSB
DNL2	Differential Non-linearity	—	V_{REF}=V_{DD}, t_{ADCK}=0.5μs Ta=-40˚C~85˚C	-4	—	+4	LSB
INL1	Integral Non-linearity	—	V_{REF}=V_{DD}, t_{ADC}=0.5μs Ta=25˚C	-4	—	+4	LSB
INL2	Integral Non-linearity	—	V_{REF}=V_{DD}, t_{ADCK}=0.5μs Ta=-40˚C~85˚C	-8	—	+8	LSB
I_{ADC}	Additional Power Consumption if A/D Converter is Used	3V	No load (t_{ADCK}=0.5μs)	—	0.9	1.35	mA
		5V	No load (t_{ADCK}=0.5μs)	—	1.2	1.8	mA
I_{BG}	Additional Power Consumption if V_{BG} Reference with Buffer is used	—	—	—	200	300	μA
t_{ADCK}	A/D Converter Clock Period	—	—	0.5	—	10	μs
t_{ADC}	A/D Conversion Time (Include Sample and Hold Time)	—	12-bit ADC	—	16	—	t_{ADCK}
t_{ADS}	A/D Converter Sampling Time	—	—	—	4	—	t_{ADCK}
t_{ON2ST}	A/D Converter On-to-Start Time	—	—	2	—	—	μs
t_{BGS}	V_{BG} Turn on Stable Time	—	—	200	—	—	μs

Comparator Electrical Characteristics

Ta=25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{CMP}	Comparator Operating Voltage	—	—	2.2	—	5.5	V
I_{CMP}	Comparator Operating Current	3V	—	—	37	56	μA
		5V	—	—	130	200	μA
V_{CMPOS}	Comparator Input Offset Voltage	—	—	−10	—	+10	mV
V_{HYS}	Hysteresis Width	—	—	20	40	60	mV
V_{CM}	Comparator Common Mode Voltage Range	—	—	V_{SS}	—	V_{DD} -1.4V	V
A_{OL}	Comparator Open Loop Gain	—	—	60	80	—	dB
t_{PD}	Comparator Response Time	—	With 100mV overdrive (Note)	—	370	560	ns

Note: Measured with comparator one input pin at V_{CM}=(V_{DD}−1.4)/2 while the other pin input transition from V_{SS} to (V_{CM} +100mV) or from V_{DD} to (V_{CM} −100mV).

RF Transceiver Electrical Characteristics

Ta=25˚C

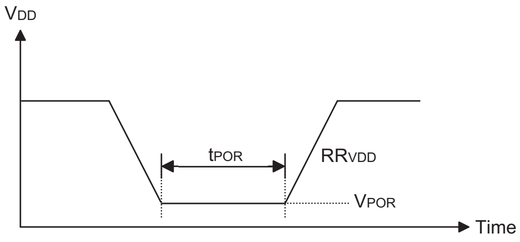
Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DDRF}	RF operating voltage (VDD3RXRF, VDD3IF, VDD3B)	—	—	1.9	3.0	3.6	V
V_{IHRF}	RF digital input high voltage	—	—	$0.7V_{DDRF}$	—	5.25	V
V_{ILRF}	RF digital input low voltage	—	—	0	—	$0.3V_{DDRF}$	V
V_{OHRF}	RF digital output high voltage	—	I=-0.25mA	V_{DDRF}-0.3V	—	V_{DDRF}	V
V_{OLRF}	RF digital output low voltage	—	I=0.25mA	0	—	0.3	V
I_{STBRF1}	RF power down current	—	—	—	4	—	μA
I_{STBRF2}	RF standby-I current	—	—	—	90	—	μA
I_{STBRF3}	RF standby-II current	—	—	—	330	—	μA
f_{OP}	RF Operating frequency	—	—	2400	—	2527	MHz
R_{FSK}	Air data rate	—	—	250	—	2000	Kbps
Transmitter							
P_{RF}	Output power	—	—	-40	0	3	dBm
PBW	Modulation 20dB bandwidth	—	R_{FSK}=2Mbps	—	2.5	—	MHz
		—	R_{FSK}=1Mbps	—	1.8	—	MHz
		—	R_{FSK}=250Kbps	—	1.6	—	MHz
I_{DDTX}	Transmitter operating current	—	P_{RF}=-35dBm	—	8	—	mA
		—	P_{RF}=-25dBm	—	9	—	mA
		—	P_{RF}=-20dBm	—	10	—	mA
		—	P_{RF}=-10dBm	—	12	—	mA
		—	P_{RF}=-6dBm	—	13	—	mA
		—	P_{RF}=-1dBm	—	18	—	mA
		—	P_{RF}=3dBm	—	25	—	mA
Receiver							
I_{DDRX}	Receiver operating current	—	R_{FSK}=2Mbps	—	18	—	mA
		—	R_{FSK}=1Mbps	—	18	—	mA
		—	R_{FSK}=250Kbps	—	18	—	mA
MaxInput	1E-3 BER	—	—	—	20	—	dBm
RXSENS	1E-3 BER sensitivity	—	R_{FSK}=2Mbps	—	-87	—	dBm
		—	R_{FSK}=1Mbps	—	-90	—	dBm
		—	R_{FSK}=250Kbps	—	-96	—	dBm
C/I CO	Co-channel C/I (2Mbps)	—	—	—	6	—	dB
C/I+1ST	ACS C/I 2MHz (2Mbps)	—	—	—	2	—	dB
C/I-1ST	ACS C/I 2MHz (2Mbps)	—	—	—	-6	—	dB
C/I+2ND	ACS C/I 4MHz (2Mbps)	—	—	—	-21	—	dB
C/I-2ND	ACS C/I 4MHz (2Mbps)	—	—	—	-12	—	dB
C/I+3RD	ACS C/I 6MHz (2Mbps)	—	—	—	-29	—	dB
C/I-3RD	ACS C/I 6MHz (2Mbps)	—	—	—	-18	—	dB

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
C/I CO	Co-channel C/I (1Mbps)	—	—	—	6	—	dB
C/I+1ST	ACS C/I 1MHz (1Mbps)	—	—	—	4	—	dB
C/I-1ST	ACS C/I 1MHz (1Mbps)	—	—	—	-6	—	dB
C/I+2ND	ACS C/I 2MHz (1Mbps)	—	—	—	-24	—	dB
C/I-2ND	ACS C/I 2MHz (1Mbps)	—	—	—	-12	—	dB
C/I+3RD	ACS C/I 3MHz (1Mbps)	—	—	—	-28	—	dB
C/I-3RD	ACS C/I 3MHz (1Mbps)	—	—	—	-16	—	dB
C/I CO	Co-channel C/I (250Kbps)	—	—	—	9	—	dB
C/I+1ST	ACS C/I 1MHz (250Kbps)	—	—	—	-13	—	dB
C/I-1ST	ACS C/I 1MHz (250Kbps)	—	—	—	-16	—	dB
C/I+2ND	ACS C/I 2MHz (250Kbps)	—	—	—	-25	—	dB
C/I-2ND	ACS C/I 2MHz (250Kbps)	—	—	—	-9	—	dB
C/I+3RD	ACS C/I 3MHz (250Kbps)	—	—	—	-33	—	dB
C/I-3RD	ACS C/I 3MHz (250Kbps)	—	—	—	-33	—	dB

Power on Reset Electrical Characteristics

Ta=25˚C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Condition				
V_{POR}	V_{DD} Start Voltage to ensure Power-on Reset	—	—	—	—	100	mV
$R_{POR\ AC}$	V_{DD} Raising Rate to Ensure Power-on Reset	—	—	0.035	—	—	V/ms
t_{POR}	Minimum Time for V_{DD} to remain at V_{POR} to ensure Power-on Reset	—	—	1	—	—	ms

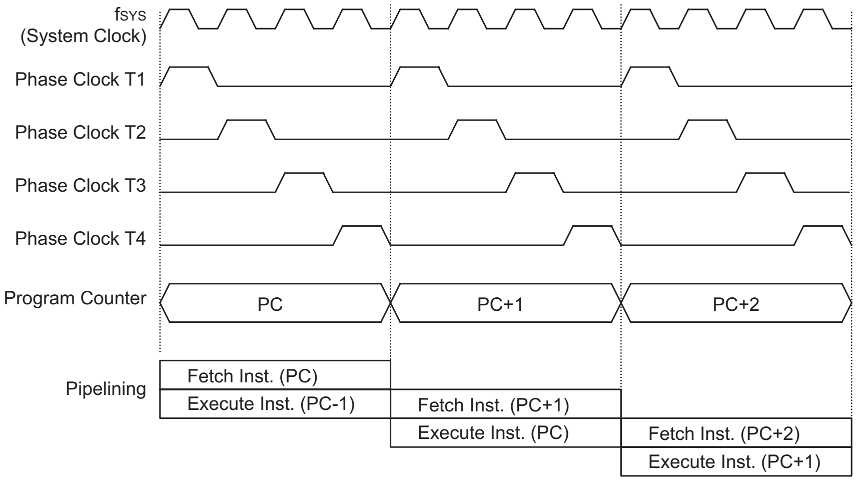


System Architecture

A key factor in the high-performance features of the Holtek range of microcontrollers is attributed to their internal system architecture. The range of devices take advantage of the usual features found within RISC microcontrollers providing increased speed of operation and enhanced performance. The pipelining scheme is implemented in such a way that instruction fetching and instruction execution are overlapped, hence instructions are effectively executed in one cycle, with the exception of branch or call instructions. An 8-bit wide ALU is used in practically all instruction set operations, which carries out arithmetic operations, logic operations, rotation, increment, decrement, branch decisions, etc. The internal data path is simplified by moving data through the Accumulator and the ALU. Certain internal registers are implemented in the Data Memory and can be directly or indirectly addressed. The simple addressing methods of these registers along with additional architectural features ensure that a minimum of external components is required to provide a functional I/O and A/D control system with maximum reliability and flexibility. This makes the devices suitable for low-cost, high-volume production for controller applications.

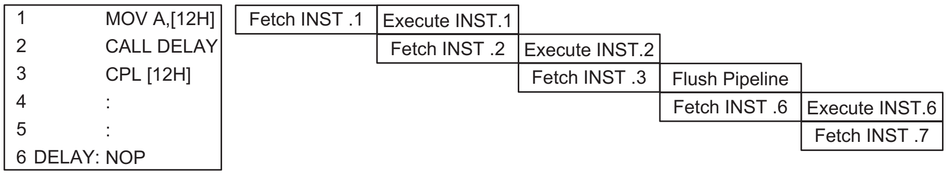
Clocking and Pipelining

The main system clock, derived from either a HXT, LXT or LIRC oscillator is subdivided into four internally generated non-overlapping clocks, T1~T4. The Program Counter is incremented at the beginning of the T1 clock during which time a new instruction is fetched. The remaining T2~T4 clocks carry out the decoding and execution functions. In this way, one T1~T4 clock cycle forms one instruction cycle. Although the fetching and execution of instructions takes place in consecutive instruction cycles, the pipelining structure of the microcontroller ensures that instructions are effectively executed in one instruction cycle. The exception to this are instructions here the contents of the Program Counter are changed, such as subroutine calls or jumps, in which case the instruction will take one more instruction cycle to execute.



System Clocking and Pipelining

For instructions involving branches, such as jump or call instructions, two machine cycles are required to complete instruction execution. An extra cycle is required as the program takes one cycle to first obtain the actual jump or call address and then another cycle to actually execute the branch. The requirement for this extra cycle should be taken into account by programmers in timing sensitive applications.



```
1          MOV A,[12H]
2          CALL DELAY
3          CPL [12H]
4          :
5          :
6 DELAY: NOP
```

Instruction Fetching

Program Counter

During program execution, the Program Counter is used to keep track of the address of the next instruction to be executed. It is automatically incremented by one each time an instruction is executed except for instructions, such as "JMP" or "CALL" that demand a jump to a non-consecutive Program Memory address. Only the lower 8 bits, known as the Program Counter Low Register, are directly addressable by the application program.

When executing instructions requiring jumps to non-consecutive addresses such as a jump instruction, a subroutine call, interrupt or reset, etc., the manages program control by loading the required address into the Program Counter. For conditional skip instructions, once the condition has been met, the next instruction, which has already been fetched during the present instruction execution, is discarded and a dummy cycle takes its place while the correct instruction is obtained.

Device	Program Counter	
	High Byte	Low Byte (PCL Register)
BC66F840	PC11~PC8	PCL7~PCL0
BC66F850	PC12~PC8	PCL7~PCL0
BC66F860	PC13~PC8	PCL7~PCL0

Program Counter

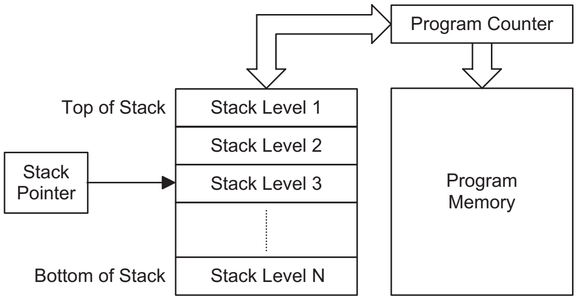
The lower byte of the Program Counter, known as the Program Counter Low register or PCL, is available for program control and is a readable and writeable register. By transferring data directly into this register, a short program jump can be executed directly, however, as only this low byte is available for manipulation, the jumps are limited to the present page of memory, that is 256 locations. When such program jumps are executed it should also be noted that a dummy cycle will be inserted. Manipulating the PCL register may cause program branching, so an extra cycle is needed to pre-fetch.

Stack

This is a special part of the memory which is used to save the contents of the Program Counter only. The stack has multiple levels depending upon the device and is neither part of the data nor part of the program space, and is neither readable nor writeable. The activated level is indexed by the Stack Pointer, and is neither readable nor writeable. At a subroutine call or interrupt acknowledge signal, the contents of the Program Counter are pushed onto the stack. At the end of a subroutine or an interrupt routine, signaled by a return instruction, RET or RETI, the Program Counter is restored to its previous value from the stack. After a device reset, the Stack Pointer will point to the top of the stack.

If the stack is full and an enabled interrupt takes place, the interrupt request flag will be recorded but the acknowledge signal will be inhibited. When the Stack Pointer is decremented, by RET or RETI, the interrupt will be serviced. This feature prevents stack overflow allowing the programmer to use the structure more easily. However, when the stack is full, a CALL subroutine instruction can still be executed which will result in a stack overflow. Precautions should be taken to avoid such cases which might cause unpredictable program branching.

If the stack is overflow, the first Program Counter save in the stack will be lost.



Device	Stack Level N
BC66F840	8
BC66F850	8
BC66F860	12

Arithmetic and Logic Unit – ALU

The arithmetic-logic unit or ALU is a critical area of the that carries out arithmetic and logic operations of the instruction set. Connected to the main data bus, the ALU receives related instruction codes and performs the required arithmetic or logical operations after which the result will be placed in the specified register. As these ALU calculation or operations may result in carry, borrow or other status changes, the status register will be correspondingly updated to reflect these changes. The ALU supports the following functions:

- Arithmetic operations: ADD, ADDM, ADC, ADCM,SUB, SUBM, SBC, SBCM, DAA
- Logic operations: AND, OR, XOR, ANDM, ORM,XORM, CPL, CPLA
- Rotation RRA, RR, RRCA, RRC, RLA, RL, RLCA, RLC
- Increment and Decrement INCA, INC, DECA, DEC
- Branch decision, JMP, SZ, SZA, SNZ, SIZ, SDZ,SIZA, SDZA, CALL, RET, RETI

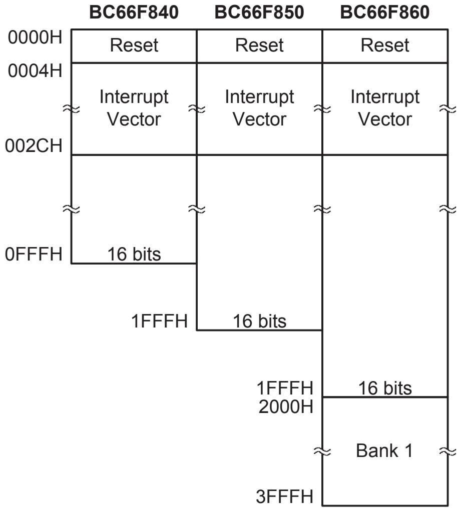
Flash Program Memory

The Program Memory is the location where the user code or program is stored. For these devices serie the Program Memory is Flash type, which means it can be programmed and re-programmed a large number of times, allowing the user the convenience of code modification on the same device. By using the appropriate programming tools, these Flash devices offer users the flexibility to conveniently debug and develop their applications while also offering a means of field programming and updating.

Structure

The Program Memory has a capacity of up to 16Kx16 bits. The Program Memory is addressed by the Program Counter and also contains data, table information and interrupt entries. Table data, which can be setup in any location within the Program Memory, is addressed by a separate table pointer register.

Device	Capacity	Program Memory Banks
BC66F840	4K×16	0
BC66F850	8K×16	0
BC66F860	16K×16	0~1



Program Memory Structure

Special Vectors

Within the Program Memory, certain locations are reserved for the reset and interrupts. The location 0000H is reserved for use by the device reset for program initialisation. After a device reset is initiated, the program will jump to this location and begin execution.

Look-up Table

Any location within the Program Memory can be defined as a look-up table where programmers can store fixed data. To use the look-up table, the table pointer must first be setup by placing the address of the look up data to be retrieved in the table pointer register, TBLP and TBHP. These registers define the total address of the look-up table.

After setting up the table pointer, the table data can be retrieved from the Program Memory using the ″TABRD[m]″ or ″TABRDL[m]″ instructions, respectively. When the instruction is executed, the lower order table byte from the Program Memory will be transferred to the user defined Data Memory register [m] as specified in the instruction. The higher order table data byte from the Program Memory will be transferred to the TBLH special register. Any unused bits in this transferred higher order byte will be read as ″0″.

The accompanying diagram illustrates the addressing data flow of the look-up table.

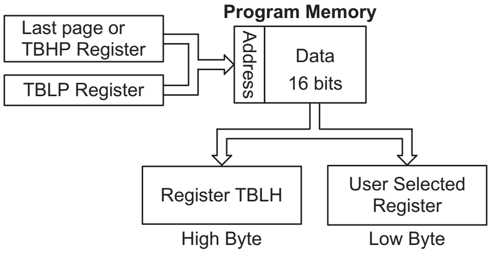


Table Program Example

The following example shows how the table pointer and table data is defined and retrieved from the microcontroller. This example uses raw table data located in the Program Memory which is stored there using the ORG statement. The value at this ORG statement is “0F00H” which refers to the start address of the last page within the 4K Program Memory of the BC66F840 device. The table pointer is setup here to have an initial value of “06H”. This will ensure that the first data read from the data table will be at the Program Memory address “0F06H” or 6 locations after the start of the last page. Note that the value for the table pointer is referenced to the first address of the present page if the “TABRD [m]” instruction is being used. The high byte of the table data which in this case is equal to zero will be transferred to the TBLH register automatically when the “TABRD [m]” instruction is executed.

Because the TBLH register is a read-only register and cannot be restored, care should be taken to ensure its protection if both the main routine and Interrupt Service Routine use table read instructions. If using the table read instructions, the Interrupt Service Routines may change the value of the TBLH and subsequently cause errors if used again by the main routine. As a rule it is recommended that simultaneous use of the table read instructions should be avoided. However, in situations where simultaneous use cannot be avoided, the interrupts should be disabled prior to the execution of any main routine table-read instructions. Note that all table related instructions require two instruction cycles to complete their operation.

Table Read Program Example:

```
tempreg1 db ?      ; temporary register #1
tempreg2 db ?      ; temporary register #2
:
:
mov  a,06h         ; initialise low table pointer - note that this address is referenced
mov  tblp,a
mov  a,0Fh         ; initialise high table pointer
mov  tbhp,a
:
:
tabrd    tempreg1  ; transfers value in table referenced by table pointer data at program
                   ; memory address "0F06H" transferred to tempreg1 and TBLH
dec  tblp          ; reduce value of table pointer by one
tabrd tempreg2     ; transfers value in table referenced by table pointer data at program
                   ; memory address "0F05H" transferred to tempreg2 and TBLH in this
                   ; example the data "1AH" is transferred to tempreg1 and data "0FH" to
                   ; register tempreg2
:
:
org  0F00h         ; sets initial address of program memory
dc   00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
:
:
```

In Circuit Programming

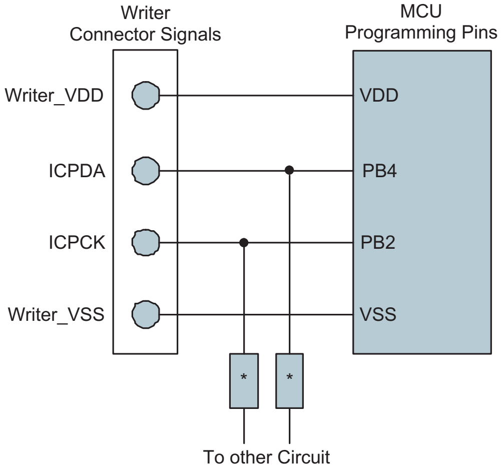
The provision of Flash type Program Memory provides the user with a means of convenient and easy upgrades and modifications to their programs on the same device.

As an additional convenience, Holtek has provided a means of programming the microcontroller in-circuit using a 4-pin interface. This provides manufacturers with the possibility of manufacturing their circuit boards complete with a programmed or un-programmed microcontroller, and then programming or upgrading the program at a later stage. This enables product manufacturers to easily keep their manufactured products supplied with the latest program releases without removal and re-insertion of the device.

Holtek Writer Pins	MCU Programming Pins	Function
ICPDA	PB4	Programming Serial Data
ICPCK	PB2	Programming Clock
VDD	VDD	Power Supply
VSS	VSS	Ground

The Program Memory can be programmed serially in-circuit using this 4-wire interface. Data is downloaded and uploaded serially on a single pin with an additional line for the clock. Two additional lines are required for the power supply and one line for the reset. The technical details regarding the in-circuit programming of the devices are beyond the scope of this document and will be supplied in supplementary literature.

During the programming process, the user must take care of the ICPDA and ICPCK pins for data and clock programming purposes to ensure that no other outputs are connected to these two pins.



Note: * may be resistor or capacitor. The resistance of * must be greater than 1kΩ or the capacitance of * must be less than 1nF.

On-Chip Debug Support – OCDS

There is an EV chip named BC66V8x0 which is used to emulate the BC66F8x0 device respectively. The BC66V8x0 device also provides the “On-Chip Debug” function to debug the corresponding BC66F8x0 device during development process. The devices, BC66F8x0 and BC66V8x0, are almost functional compatible except the “On-Chip Debug” function. Users can use the BC66V8x0 device to emulate the BC66F8x0 device behaviors by connecting the OCDSDA and OCDSCK pins to the Holtek HT-IDE development tools. The OCDSDA pin is the OCDS Data/Address input/output pin while the OCDSCK pin is the OCDS clock input pin. When users use the BC66V8x0 EV chip for debugging, the corresponding pin functions shared with the OCDSDA and OCDSCK pins in the BC66F8x0 device will have no effect in the BC66V8x0 EV chip. However, the two OCDS pins which are pin-shared with the ICP programming pins are still used as the Flash Memory programming pins for ICP. For more detailed OCDS information, refer to the corresponding document named “Holtek e-Link for 8-bit MCU OCDS User’s Guide”.

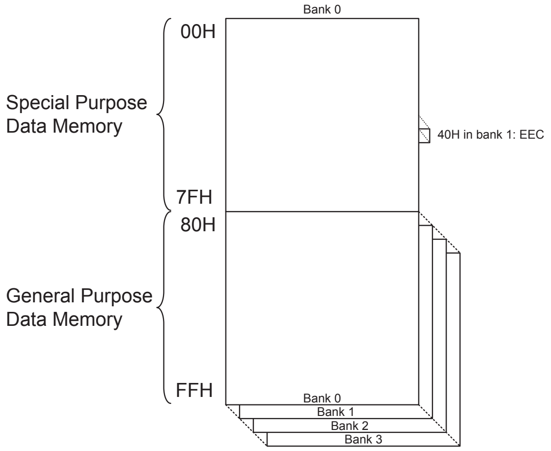
Holtek e-Link Pins	EV Chip Pins	Pin Description
OCDSDA	OCDSDA	On-Chip Debug Support Data/Address input/output
OCDSCK	OCDSCK	On-Chip Debug Support Clock input
VDD	VDD	Power Supply
GND	VSS	Ground

RAM Data Memory

The Data Memory is a volatile area of 8-bit wide RAM internal memory and is the location where temporary information is stored.

Structure

Divided into two sections, the first of these is an area of RAM, known as the Special Function Data Memory. Here are located registers which are necessary for correct operation of the device. Many of these registers can be read from and written to directly under program control, however, some remain protected from user manipulation.



For BS66F840, there are 2 Data memory banks.
For BS66F850, there are 3 Data memory banks.
For BS66F860, there are 4 Data memory banks.

Data Memory Structure

General Purpose Data Memory

All microcontroller programs require an area of read/write memory where temporary data can be stored and retrieved for use later. It is this area of RAM memory that is known as General Purpose Data Memory. This area of Data Memory is fully accessible by the user programming for both reading and writing operations. By using the bit operation instructions individual bits can be set or reset under program control giving the user a large range of flexibility for bit manipulation in the Data Memory.

Device	Capacity	Banks
BC66F840	256×8	0: 80H~FFH 1: 80H~FFH
BC66F850	384×8	0: 80H~FFH 1: 80H~FFH 2: 80H~FFH
BC66F860	512×8	0: 80H~FFH 1: 80H~FFH 2: 80H~FFH 3: 80H~FFH

General Purpose Data Memory Structure

Special Purpose Data Memory

This area of Data Memory is where registers, necessary for the correct operation of the microcontroller, are stored. Most of the registers are both readable and writeable but some are protected and are readable only, the details of which are located under the relevant Special Function Register section. Note that for locations that are unused, any read instruction to these addresses will return the value "00H".

	Bank 0~1
00H	IAR0
01H	MP0
02H	IAR1
03H	MP1
04H	BP
05H	ACC
06H	PCL
07H	TBLP
08H	TBLH
09H	TBHP
0AH	STATUS
0BH	SMOD
0CH	LVDC
0DH	INTEG
0EH	INTC0
0FH	INTC1
10H	INTC2
11H	MFI0
12H	Unused
13H	MFI2
14H	MFI3
15H	MFI4
16H	PA
17H	PAC
18H	PAPU
19H	PAWU
1AH	Unused
1BH	TMPC
1CH	WDTC
1DH	TBC
1EH	LVRC
1FH	CPC
20H	ADRL
21H	ADRH
22H	ADCR0
23H	ADCR1
24H	ACERL
25H	Unused
26H	CTRL0
27H	CTRL1
28H	TM0C0
29H	TM0C1
2AH	TM0DL
2BH	TM0DH
2CH	TM0AL
2DH	TM0AH
2EH	TM0RP
2FH ~ 35H	Unused

	Bank 0	Bank 1
36H	TM2C0	
37H	TM2C1	
38H	TM2DL	
39H	TM2DH	
3AH	TM2AL	
3BH	TM2AH	
3CH	TM2RP	
3DH	Unused	
3EH	EEA	
3FH	EED	
40H	Unused	EEC
41H	Unused	
42H	TM3C0	
43H	TM3C1	
44H	TM3C2	
45H	TM3DL	
46H	TM3DH	
47H	TM3AL	
48H	TM3AH	
49H	TM3BL	
4AH	TM3BH	
4BH	TM3RP	
4CH	Unused	
4DH	PB	
4EH	PBC	
4FH	PBPU	
50H	PC	
51H	PCC	
52H	PCPU	
53H	PD	
54H	PDC	
55H	PDPU	
56H ~ 5FH	Unused	
60H	I2CTOC	
61H	SIMC0	
62H	SIMC1	
63H	SIMD	
64H	SIMA/SIMC2	
65H	SPIAC0	
66H	SPIAC1	
67H	SPIAD	
68H ~ 7FH	Unused	

▨ : Unused, read as 00H

Special Purpose Data Memory Structure – BC66F840

Address	Bank 0~2
00H	IAR0
01H	MP0
02H	IAR1
03H	MP1
04H	BP
05H	ACC
06H	PCL
07H	TBLP
08H	TBLH
09H	TBHP
0AH	STATUS
0BH	SMOD
0CH	LVDC
0DH	INTEG
0EH	INTC0
0FH	INTC1
10H	INTC2
11H	MFI0
12H	MFI1
13H	MFI2
14H	MFI3
15H	MFI4
16H	PA
17H	PAC
18H	PAPU
19H	PAWU
1AH	Unused
1BH	TMPC
1CH	WDTC
1DH	TBC
1EH	LVRC
1FH	CPC
20H	ADRL
21H	ADRH
22H	ADCR0
23H	ADCR1
24H	ACERL
25H	ACERH
26H	CTRL0
27H	CTRL1
28H	TM0C0
29H	TM0C1
2AH	TM0DL
2BH	TM0DH
2CH	TM0AL
2DH	TM0AH
2EH	TM0RP
2FH	TM1C0
30H	TM1C1
31H	TM1DL
32H	TM1DH
33H	TM1AL
34H	TM1AH
35H	TM1RP

Address	Bank 0, 2	Bank 1
36H	TM2C0	
37H	TM2C1	
38H	TM2DL	
39H	TM2DH	
3AH	TM2AL	
3BH	TM2AH	
3CH	TM2RP	
3DH	Unused	
3EH	EEA	
3FH	EED	
40H	Unused	EEC
41H	Unused	
42H	TM3C0	
43H	TM3C1	
44H	TM3C2	
45H	TM3DL	
46H	TM3DH	
47H	TM3AL	
48H	TM3AH	
49H	TM3BL	
4AH	TM3BH	
4BH	TM3RP	
4CH	Unused	
4DH	PB	
4EH	PBC	
4FH	PBPU	
50H	PC	
51H	PCC	
52H	PCPU	
53H	PD	
54H	PDC	
55H	PDPU	
56H	PE	
57H	PEC	
58H	PEPU	
59H 5FH	≈ Unused ≈	
60H	I2CTOC	
61H	SIMC0	
62H	SIMC1	
63H	SIMD	
64H	SIMA/SIMC2	
65H	SPIAC0	
66H	SPIAC1	
67H	SPIAD	
68H 7FH	≈ Unused ≈	

▨ : Unused, read as 00H

Special Purpose Data Memory Structure – BC66F850

	Bank 0~3		Bank 0, 2~3	Bank 1
00H	IAR0	36H	TM2C0	
01H	MP0	37H	TM2C1	
02H	IAR1	38H	TM2DL	
03H	MP1	39H	TM2DH	
04H	BP	3AH	TM2AL	
05H	ACC	3BH	TM2AH	
06H	PCL	3CH	TM2RP	
07H	TBLP	3DH	Unused	
08H	TBLH	3EH	EEA	
09H	TBHP	3FH	EED	
0AH	STATUS	40H	Unused	EEC
0BH	SMOD	41H	Unused	
0CH	LVDC	42H	TM3C0	
0DH	INTEG	43H	TM3C1	
0EH	INTC0	44H	TM3C2	
0FH	INTC1	45H	TM3DL	
10H	INTC2	46H	TM3DH	
11H	MFI0	47H	TM3AL	
12H	MFI1	48H	TM3AH	
13H	MFI2	49H	TM3BL	
14H	MFI3	4AH	TM3BH	
15H	MFI4	4BH	TM3RP	
16H	PA	4CH	Unused	
17H	PAC	4DH	PB	
18H	PAPU	4EH	PBC	
19H	PAWU	4FH	PBPU	
1AH	Unused	50H	PC	
1BH	TMPC	51H	PCC	
1CH	WDTC	52H	PCPU	
1DH	TBC	53H	PD	
1EH	LVRC	54H	PDC	
1FH	CPC	55H	PDPU	
20H	ADRL	56H	PE	
21H	ADRH	57H	PEC	
22H	ADCR0	58H	PEPU	
23H	ADCR1	59H	PF	
24H	ACERL	5AH	PFC	
25H	ACERH	5BH	PFPU	
26H	CTRL0	5CH 5FH	≈ Unused ≈	
27H	CTRL1			
28H	TM0C0			
29H	TM0C1	60H	I2CTOC	
2AH	TM0DL	61H	SIMC0	
2BH	TM0DH	62H	SIMC1	
2CH	TM0AL	63H	SIMD	
2DH	TM0AH	64H	SIMA/SIMC2	
2EH	TM0RP	65H	SPIAC0	
2FH	TM1C0	66H	SPIAC1	
30H	TM1C1	67H	SPIAD	
31H	TM1DL	68H 7FH	≈ Unused ≈	
32H	TM1DH			
33H	TM1AL			
34H	TM1AH			
35H	TM1RP			

▨ : Unused, read as 00H

Special Purpose Data Memory Structure – BC66F860

Special Function Register Description

Most of the Special Function Register details will be described in the relevant functional section, however several registers require a separate description in this section.

Indirect Addressing Registers – IAR0, IAR1

The Indirect Addressing Registers, IAR0 and IAR1, although having their locations in normal RAM register space, do not actually physically exist as normal registers. The method of indirect addressing for RAM data manipulation uses these Indirect Addressing Registers and Memory Pointers, in contrast to direct memory addressing, where the actual memory address is specified. Actions on the IAR0 and IAR1 registers will result in no actual read or write operation to these registers but rather to the memory location specified by their corresponding Memory Pointers, MP0 or MP1. Acting as a pair, IAR0 and MP0 can together access data from Bank 0 while the IAR1 and MP1 register pair can access data from any bank. As the Indirect Addressing Registers are not physically implemented, reading the Indirect Addressing Registers indirectly will return a result of "00H" and writing to the registers indirectly will result in no operation.

Memory Pointers – MP0, MP1

Two Memory Pointers, known as MP0 and MP1 are provided. These Memory Pointers are physically implemented in the Data Memory and can be manipulated in the same way as normal registers providing a convenient way with which to address and track data. When any operation to the relevant Indirect Addressing Registers is carried out, the actual address that the microcontroller is directed to, is the address specified by the related Memory Pointer. MP0, together with Indirect Addressing Register, IAR0, are used to access data from Bank 0, while MP1 and IAR1 are used to access data from all banks according to BP register. Direct Addressing can only be used with Bank 0, all other Banks must be addressed indirectly using MP1 and IAR1. Note that for this series of devices, the Memory Pointers, MP0 and MP1, are both 8-bit registers and used to access the Data Memory together with their corresponding indirect addressing registers IAR0 and IAR1.

The following example shows how to clear a section of four Data Memory locations already defined as locations adres1 to adres4.

Indirect Addressing Program Example

```
data .section data
adres1 db ?
adres2 db ?
adres3 db ?
adres4 db ?
block db ?
code .section at 0 code
org 00h
start:
mov a,04h              ; setup size of block
mov block,a
mov a,offset adres1 ; Accumulator loaded with first RAM address
mov mp0,a              ; setup memory pointer with first RAM address
loop:
clr IAR0               ; clear the data at address defined by MP0
inc mp0                ; increment memory pointer
sdz block              ; check if last memory location has been cleared
jmp loop
continue:
```

The important point to note here is that in the example shown above, no reference is made to specific RAM addresses.

Bank Pointer – BP

For this series of devices, the Data Memory is divided into up to four banks depending upon which device is selected while the Program Memory is divided into two banks for the BC66F860 device. Selecting the required Data or Program Memory area is achieved using the Bank Pointer. Bit 1~0 are used to select Data Memory Banks while the bit 5 is used to select Program Memory Banks.

The Data Memory is initialised to Bank 0 after a reset, except for a WDT time-out reset in the Power Down Mode, in which case, the Data Memory bank remains unaffected. It should be noted that the Special Function Data Memory is not affected by the bank selection, which means that the Special Function Registers can be accessed from within any bank. Directly addressing the Data Memory will always result in Bank 0 being accessed irrespective of the value of the Bank Pointer. Accessing data from banks other than Bank 0 must be implemented using indirect addressing.

Device	Bit							
	7	6	5	4	3	2	1	0
BC66F840	—	—	—	—	—	—	—	DMBP0
BC66F850	—	—	—	—	—	—	DMBP1	DMBP0
BC66F860	—	—	PMBP0	—	—	—	DMBP1	DMBP0

BP Register List

BC66F840 BP Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	—	DMBP0
R/W	—	—	—	—	—	—	—	R/W
POR	—	—	—	—	—	—	—	0

Bit 7~1 Unimplemented, read as "0"

Bit 0 **DMBP0:** Data Memory Bank Pointer
0: Bank 0
1: Bank 1

BC66F850 BP Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	DMBP1	DMBP0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **DMBP0:** Data Memory Bank Pointer
00: Bank 0
01: Bank 1
10: Bank 2
11: Bank 2

BC66F860 BP Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	PMBP0	—	—	—	DMBP1	DMBP0
R/W	—	—	R/W	—	—	—	R/W	R/W
POR	—	—	0	—	—	—	0	0

Bit 7~6 Unimplemented, read as “0”

Bit 5 **PMBP0:** Program Memory Bank Pointer
0: Bank 0, Program Memory Address is from 0000H~1FFFH
1: Bank 1, Program Memory Address is from 2000H~3FFFH

Bit 4~2 Unimplemented, read as “0”

Bit 1~0 **DMBP0:** Data Memory Bank Pointer
00: Bank 0
01: Bank 1
10: Bank 2
11: Bank 3

Accumulator – ACC

The Accumulator is central to the operation of any microcontroller and is closely related with operations carried out by the ALU. The Accumulator is the place where all intermediate results from the ALU are stored. Without the Accumulator it would be necessary to write the result of each calculation or logical operation such as addition, subtraction, shift, etc., to the Data Memory resulting in higher programming and timing overheads. Data transfer operations usually involve the temporary storage function of the Accumulator; for example, when transferring data between one user defined register and another, it is necessary to do this by passing the data through the Accumulator as no direct transfer between two registers is permitted.

Program Counter Low Register – PCL

To provide additional program control functions, the low byte of the Program Counter is made accessible to programmers by locating it within the Special Purpose area of the Data Memory. By manipulating this register, direct jumps to other program locations are easily implemented. Loading a value directly into this PCL register will cause a jump to the specified Program Memory location, however, as the register is only 8-bit wide, only jumps within the current Program Memory page are permitted. When such operations are used, note that a dummy cycle will be inserted.

Look-up Table Registers – TBLP, TBHP, TBLH

These three special function registers are used to control operation of the look-up table which is stored in the Program Memory. TBLP and TBHP are the table pointer and indicates the location where the table data is located. Their value must be setup before any table read commands are executed. Their value can be changed, for example using the "INC" or "DEC" instructions, allowing for easy table data pointing and reading. TBLH is the location where the high order byte of the table data is stored after a table read data instruction has been executed. Note that the lower order table data byte is transferred to a user defined location.

Status Register – STATUS

This 8-bit register contains the zero flag (Z), carry flag (C), auxiliary carry flag (AC), overflow flag (OV), power down flag (PDF), and watchdog time-out flag (TO). These arithmetic/logical operation and system management flags are used to record the status and operation of the microcontroller.

With the exception of the TO and PDF flags, bits in the status register can be altered by instructions like most other registers. Any data written into the status register will not change the TO or PDF flag. In addition, operations related to the status register may give different results due to the different instruction operations. The TO flag can be affected only by a system power-up, a WDT time-out or by executing the ″CLR WDT″ or ″HALT″ instruction. The PDF flag is affected only by executing the ″HALT″ or ″CLR WDT″ instruction or during a system power-up.

The Z, OV, AC and C flags generally reflect the status of the latest operations.

- **C** is set if an operation results in a carry during an addition operation or if a borrow does not take place during a subtraction operation; otherwise C is cleared. C is also affected by a rotate through carry instruction.
- **AC** is set if an operation results in a carry out of the low nibbles in addition, or no borrow from the high nibble into the low nibble in subtraction; otherwise AC is cleared.
- **Z** is set if the result of an arithmetic or logical operation is zero; otherwise Z is cleared.
- **OV** is set if an operation results in a carry into the highest-order bit but not a carry out of the highest-order bit, or vice versa; otherwise OV is cleared.
- **PDF** is cleared by a system power-up or executing the “CLR WDT” instruction. PDF is set by executing the “HALT” instruction.
- **TO** is cleared by a system power-up or executing the “CLR WDT” or “HALT” instruction. TO is set by a WDT time-out.

In addition, on entering an interrupt sequence or executing a subroutine call, the status register will not be pushed onto the stack automatically. If the contents of the status registers are important and if the subroutine can corrupt the status register, precautions must be taken to correctly save it.

STATUS Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	TO	PDF	OV	Z	AC	C
R/W	—	—	R	R	R/W	R/W	R/W	R/W
POR	—	—	0	0	x	x	x	x

"x" unknown

Bit 7, 6 Unimplemented, read as "0"

Bit 5 **TO:** Watchdog Time-Out flag
0: After power up or executing the "CLR WDT" or "HALT" instruction
1: A watchdog time-out occurred.

Bit 4 **PDF:** Power down flag
0: After power up or executing the "CLR WDT" instruction
1: By executing the "HALT" instruction

Bit 3 **OV:** Overflow flag
0: No overflow
1: An operation results in a carry into the highest-order bit but not a carry out of the highest-order bit or vice versa.

Bit 2 **Z:** Zero flag
0: The result of an arithmetic or logical operation is not zero
1: The result of an arithmetic or logical operation is zero

Bit 1 **AC:** Auxiliary flag
0: No auxiliary carry
1: An operation results in a carry out of the low nibbles in addition, or no borrow from the high nibble into the low nibble in subtraction

Bit 0 **C:** Carry flag
0: No carry-out
1: An operation results in a carry during an addition operation or if a borrow does not take place during a subtraction operation

C is also affected by a rotate through carry instruction.

EEPROM Data Memory

The devices contain an area of internal EEPROM Data Memory. EEPROM, which stands for Electrically Erasable Programmable Read Only Memory, is by its nature a non-volatile form of re-programmable memory, with data retention even when its power supply is removed. By incorporating this kind of data memory, a whole new host of application possibilities are made available to the designer. The availability of EEPROM storage allows information such as product identification numbers, calibration values, specific user data, system setup data or other product information to be stored directly within the product microcontroller. The process of reading and writing data to the EEPROM memory has been reduced to a very trivial affair.

EEPROM Data Memory Structure

The EEPROM Data Memory capacity is up to 256×8 bits for this series of devices. Unlike the Program Memory and RAM Data Memory, the EEPROM Data Memory is not directly mapped into memory space and is therefore not directly addressable in the same way as the other types of memory. Read and Write operations to the EEPROM are carried out in single byte operations using an address and data register in Bank 0 and a single control register in Bank 1.

Device	Capacity	Address
BC66F840	128×8	00H~7FH
BC66F850 BC66F860	256×8	00H~FFH

EEPROM Data Memory Structure

EEPROM Registers

Three registers control the overall operation of the internal EEPROM Data Memory. These are the address register, EEA, the data register, EED and a single control register, EEC. As both the EEA and EED registers are located in Bank 0, they can be directly accessed in the same was as any other Special Function Register. The EEC register however, being located in Bank1, cannot be addressed directly and can only be read from or written to indirectly using the MP1 Memory Pointer and Indirect Addressing Register, IAR1. Because the EEC control register is located at address 40H in Bank 1, the MP1 Memory Pointer must first be set to the value 40H and the Bank Pointer register, BP, set to the value, 01H, before any operations on the EEC register are executed.

Name		Bit							
		7	6	5	4	3	2	1	0
EEA	BC66F840	—	A6	A5	A4	A3	A2	A1	A0
	BC66F850 BC66F860	A7	A6	A5	A4	A3	A2	A1	A0
EED (All devices)		D7	D6	D5	D4	D3	D2	D1	D0
EEC (All devices)		—	—	—	—	WREN	WR	RDEN	RD

EEPROM Register List

EEA Register – BC66F840

Bit	7	6	5	4	3	2	1	0
Name	—	A6	A5	A4	A3	A2	A1	A0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	x	x	x	x	x	x	x

"x": unknown

Bit 7 Unimplemented, read as "0"

Bit 6~0 Data EEPROM Memory address
Data EEPROM Memory address bit 6~bit 0

EEA Register – BC66F850/BC66F860

Bit	7	6	5	4	3	2	1	0
Name	A7	A6	A5	A4	A3	A2	A1	A0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x": unknown

Bit 7~0 Data EEPROM Memory address
Data EEPROM Memory address bit 7~bit 0

EED Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x": unknown

Bit 7~0 EEPROM Data bits

EEC Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	WREN	WR	RDEN	RD
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4 Unimplemented, read as "0"

Bit 3 **WREN:** Data EEPROM Write Enable
0: Disable
1: Enable

This is the Data EEPROM Write Enable Bit which must be set high before Data EEPROM write operations are carried out. Clearing this bit to zero will inhibit Data EEPROM write operations.

Bit 2 **WR:** EEPROM Write Control
0: Write cycle has finished
1: Activate a write cycle

This is the Data EEPROM Write Control Bit and when set high by the application program will activate a write cycle. This bit will be automatically reset to zero by the hardware after the write cycle has finished. Setting this bit high will have no effect if the WREN has not first been set high.

Bit 1 **RDEN:** Data EEPROM Read Enable
0: Disable
1: Enable

This is the Data EEPROM Read Enable Bit which must be set high before Data EEPROM read operations are carried out. Clearing this bit to zero will inhibit Data EEPROM read operations.

Bit 0 **RD** : EEPROM Read Control
0: Read cycle has finished
1: Activate a read cycle

This is the Data EEPROM Read Control Bit and when set high by the application program will activate a read cycle. This bit will be automatically reset to zero by the hardware after the read cycle has finished. Setting this bit high will have no effect if the RDEN has not first been set high.

Note: The WREN, WR, RDEN and RD can not be set to "1" at the same time in one instruction. The WR and RD can not be set to "1" at the same time.

Reading Data from the EEPROM

To read data from the EEPROM, the read enable bit, RDEN, in the EEC register must first be set high to enable the read function. The EEPROM address of the data to be read must then be placed in the EEA register. If the RD bit in the EEC register is now set high, a read cycle will be initiated. Setting the RD bit high will not initiate a read operation if the RDEN bit has not been set. When the read cycle terminates, the RD bit will be automatically cleared to zero, after which the data can be read from the EED register. The data will remain in the EED register until another read or write operation is executed. The application program can poll the RD bit to determine when the data is valid for reading.

Writing Data to the EEPROM

The EEPROM address of the data to be written must first be placed in the EEA register and the data placed in the EED register. To write data to the EEPROM, the write enable bit, WREN, in the EEC register must first be set high to enable the write function. After this, the WR bit in the EEC register must be immediately set high to initiate a write cycle. These two instructions must be executed consecutively. The global interrupt bit EMI should also first be cleared before implementing any write operations, and then set again after the write cycle has started. Note that setting the WR bit high will not initiate a write cycle if the WREN bit has not been set. As the EEPROM write cycle is controlled using an internal timer whose operation is asynchronous to microcontroller system clock, a certain time will elapse before the data will have been written into the EEPROM. Detecting when the write cycle has finished can be implemented either by polling the WR bit in the EEC register or by using the EEPROM interrupt. When the write cycle terminates, the WR bit will be automatically cleared to zero by the microcontroller, informing the user that the data has been written to the EEPROM. The application program can therefore poll the WR bit to determine when the write cycle has ended.

Write Protection

Protection against inadvertent write operation is provided in several ways. After the devices are powered-on the Write Enable bit in the control register will be cleared preventing any write operations. Also at power-on the Bank Pointer, BP, will be reset to zero, which means that Data Memory Bank 0 will be selected. As the EEPROM control register is located in Bank 1, this adds a further measure of protection against spurious write operations. During normal program operation, ensuring that the Write Enable bit in the control register is cleared will safeguard against incorrect write operations.

EEPROM Interrupt

The EEPROM write interrupt is generated when an EEPROM write cycle has ended. The EEPROM interrupt must first be enabled by setting the DEE bit in the relevant interrupt register. However as the EEPROM is contained within a Multi-function Interrupt, the associated Multi-function Interrupt enable bit must also be set. When an EEPROM write cycle ends, the DEF request flag and its associated Multi-function Interrupt request flag will both be set. If the global, EEPROM and Multi-function interrupts are enabled and the stack is not full, a jump to the associated Multi-function Interrupt vector will take place. When the interrupt is serviced only the Multi-function interrupt flag will be automatically reset, the EEPROM interrupt flag must be manually reset by the application program. More details can be obtained in the Interrupt section.

Programming Considerations

Care must be taken that data is not inadvertently written to the EEPROM. Protection can be enhanced by ensuring that the Write Enable bit is normally cleared to zero when not writing. Also the Bank Pointer could be normally cleared to zero as this would inhibit access to Bank 1 where the EEPROM control register exist. Although certainly not necessary, consideration might be given in the application program to the checking of the validity of new write data by a simple read back process. When writing data the WR bit must be set high immediately after the WREN bit has been set high, to ensure the write cycle executes correctly. The global interrupt bit EMI should also be cleared before a write cycle is executed and then re-enabled after the write cycle starts.

Programming Examples

Reading data from the EEPROM – polling method

```
MOV A, EEPROM_ADRES    ; user defined address
MOV EEA, A
MOV A, 040H            ; setup memory pointer MP1
MOV MP1, A             ; MP1 points to EEC register
MOV A, 01H             ; setup Bank Pointer
MOV BP, A
SET IAR1.1             ; set RDEN bit, enable read operations
SET IAR1.0             ; start Read Cycle - set RD bit
BACK:
SZ IAR1.0              ; check for read cycle end
JMP BACK
CLR IAR1               ; disable EEPROM read/write
CLR BP
MOV A, EED             ; move read data to register
MOV READ_DATA, A
```

Writing Data to the EEPROM – polling method

```
CLR EMI
MOV A, EEPROM_ADRES    ; user defined address
MOV EEA, A
MOV A, EEPROM_DATA     ; user defined data
MOV EED, A
MOV A, 040H            ; setup memory pointer MP1
MOV MP1, A             ; MP1 points to EEC register
MOV A, 01H             ; setup Bank Pointer
MOV BP, A
SET IAR1.3             ; set WREN bit, enable write operations
SET IAR1.2             ; Start Write Cycle - set WR bit - executed immediately
                       ; after set WREN bit
SET EMI
BACK:
SZ IAR1.2              ; check for write cycle end
JMP BACK
CLR IAR1               ; disable EEPROM read/write
CLR BP
```

Oscillator

Various oscillator options offer the user a wide range of functions according to their various application requirements. The flexible features of the oscillator functions ensure that the best optimisation can be achieved in terms of speed and power saving. Oscillator selections and operation are selected through a combination of configuration options and registers.

Oscillator Overview

In addition to being the source of the main system clock the oscillators also provide clock sources for theWatchdog Timer and Time Base Interrupts. External oscillators requiring some external components as well as fully integrated internal oscillators, requiring no external components, are provided to form a wide range of both fast and slow system oscillators. All oscillator options are selected through the configuration options. The higher frequency oscillators provide higher performance but carry with it the disadvantage of higher power requirements, while the opposite is of course true for the lower frequency oscillators. With the capability of dynamically switching between fast and slow system clock, the devices have the flexibility to optimize the performance/power ratio, a feature especially important in power sensitive portable applications.

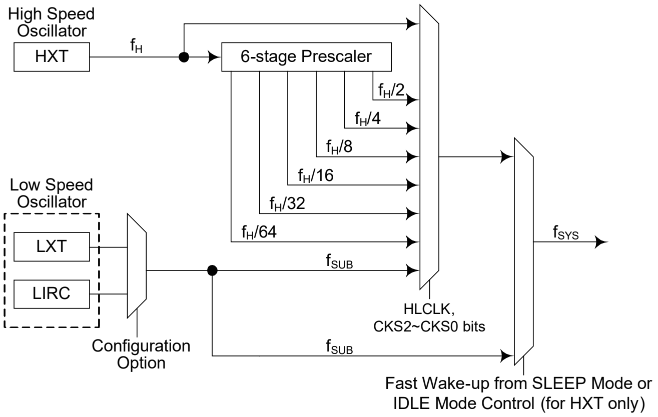
Type	Name	Freq.	Pins
External Crystal	HXT	16MHz	OSC1/OSC2
External Low Speed Crystal	LXT	32.768kHz	XT1/XT2
Internal Low Speed RC	LIRC	32kHz	—

Oscillator Types

System Clock Configurations

There are three system oscillators, one high speed oscillator and two low speed oscillators. The high speed oscillator is the external crystal/ceramic oscillator – HXT. The low speed oscillators are the internal 32kHz oscillator – LIRC and the external 32.768kHz crystal oscillator. Selecting whether the low or high speed oscillator is used as the system oscillator is implemented using the HLCLK bit and CKS2~CKS0 bits in the SMOD register and as the system clock can be dynamically selected.

The actual source clock used for the low speed oscillator is chosen via a configuration option. The frequency of the slow speed or high speed system clock is also determined using the HLCLK bit and CKS2~CKS0 bits in the SMOD register. Note that two oscillator selections must be made namely one high speed and one low speed system oscillators. It is not possible to choose a no-oscillator selection for either the high or low speed oscillator.

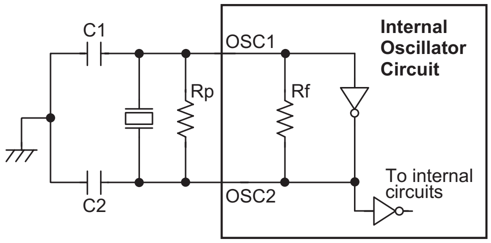


System Clock Configurations

External Crystal/Ceramic Oscillator – HXT

The External Crystal/Ceramic System Oscillator is the high frequency oscillator. For most crystal oscillator configurations, the simple connection of a crystal across OSC1 and OSC2 will create the necessary phase shift and feedback for oscillation, without requiring external capacitors. However, for some crystal types and frequencies, to ensure oscillation, it may be necessary to add two small value capacitors, C1 and C2. Using a ceramic resonator will usually require two small value capacitors, C1 and C2, to be connected as shown for oscillation to occur. The values of C1 and C2 should be selected in consultation with the crystal or resonator manufacturers specification.

For oscillator stability and to minimise the effects of noise and crosstalk, it is important to ensure that the crystal and any associated resistors and capacitors along with inter connecting lines are all located as close to the MCU as possible.



Note: 1. Rp is normally not required. C1 and C2 are required.
2. Although not shown OSC1/OSC2 pins have a parasitic capacitance of around 7pF.

Crystal/Resonator Oscillator – HXT

Crystal Oscillator C1 and C2 Values		
Crystal Frequency	**C1**	**C2**
16MHz	0pF	0pF
Note: C1 and C2 values are for guidance only.		

Crystal Recommended Capacitor Values

External 32.768kHz Crystal Oscillator – LXT

The External 32.768kHz Crystal System Oscillator is one of the low frequency oscillator choices, which is selected via configuration option. This clock source has a fixed frequency of 32.768kHz and requires a 32.768kHz crystal to be connected between pins XT1 and XT2. The external resistor and capacitor components connected to the 32.768kHz crystal are necessary to provide oscillation. For applications where precise frequencies are essential, these components may be required to provide frequency compensation due to different crystal manufacturing tolerances. During power-up there is a time delay associated with the LXT oscillator waiting for it to start-up.

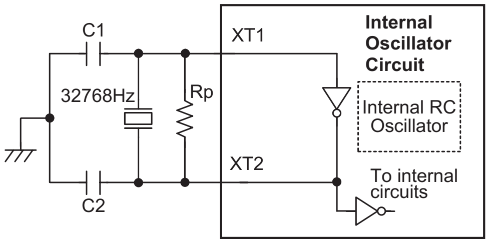
When the microcontroller enters the SLEEP or IDLE Mode, the system clock is switched off to stop microcontroller activity and to conserve power. However, in many microcontroller applications it may be necessary to keep the internal timers operational even when the microcontroller is in the SLEEP or IDLE Mode. To do this, another clock, independent of the system clock, must be provided.

However, for some crystals, to ensure oscillation and accurate frequency generation, it is necessary to add two small value external capacitors, C1 and C2. The exact values of C1 and C2 should be selected in consultation with the crystal or resonator manufacturer's specification. The external parallel feedback resistor, Rp, is required.

A configuration option determines if the XT1/XT2 pins are used for the LXT oscillator or as I/O pins.

- If the LXT oscillator is not used for any clock source, the XT1/XT2 pins can be used as normal I/O pins.
- If the LXT oscillator is used for any clock source, the 32.768kHz crystal should be connected to the XT1/XT2 pins.

For oscillator stability and to minimise the effects of noise and crosstalk, it is important to ensure that the crystal and any associated resistors and capacitors along with inter connecting lines are all located as close to the MCU as possible.



Note: 1. Rp, C1 and C2 are required.
2. Although not shown pins have a parasitic capacitance of around 7pF.

External LXT Oscillator – LXT

LXT Oscillator C1 and C2 Values		
Crystal Frequency	**C1**	**C2**
32.768kHz	10pF	10pF
Note: 1. C1 and C2 values are for guidance only. 2. Rp=5M~10MΩ is recommended.		

32.768kHz Crystal Recommended Capacitor Values

LXT Oscillator Low Power Function

The LXT oscillator can function in one of two modes, the Quick Start Mode and the Low Power Mode. The mode selection is executed using the LXTLP bit in the TBC register.

LXTLP Bit	LXT Mode
0	Quick Start
1	Low Power

After power on, the LXTLP bit will be automatically cleared to zero ensuring that the LXT oscillator is in the Quick Start operating mode. In the Quick Start Mode the LXT oscillator will power up and stabilise quickly. However, after the LXT oscillator has fully powered up it can be placed into the Low-power mode by setting the LXTLP bit high. The oscillator will continue to run but with reduced current consumption, as the higher current consumption is only required during the LXT oscillator start-up. In power sensitive applications, such as battery applications, where power consumption must be kept to a minimum, it is therefore recommended that the application program sets the LXTLP bit high about 2 seconds after power-on.

It should be noted that no matter what condition the LXTLP bit is set to, the LXT oscillator will always function normally, the only difference is that it will take more time to start up if in the Low-power mode.

Internal 32kHz Oscillator – LIRC

The Internal 32kHz System Oscillator is the low frequency oscillator. It is a fully integrated RC oscillator with a typical frequency of 32kHz at 5V, requiring no external components for its implementation. Device trimming during the manufacturing process and the inclusion of internal frequency compensation circuits are used to ensure that the influence of the power supply voltage, temperature and process variations on the oscillation frequency are minimised. As a result, at a power supply of 5V and at a temperature of 25˚C degrees, the fixed oscillation frequency of 32kHz will have a tolerance within 10%.

Supplementary Oscillator

The low speed oscillator, in addition to providing a system clock source, is also used to provide a clock source to other device functions such as the Watchdog Timer and the Time Base Interrupts.

Operating Modes and System Clocks

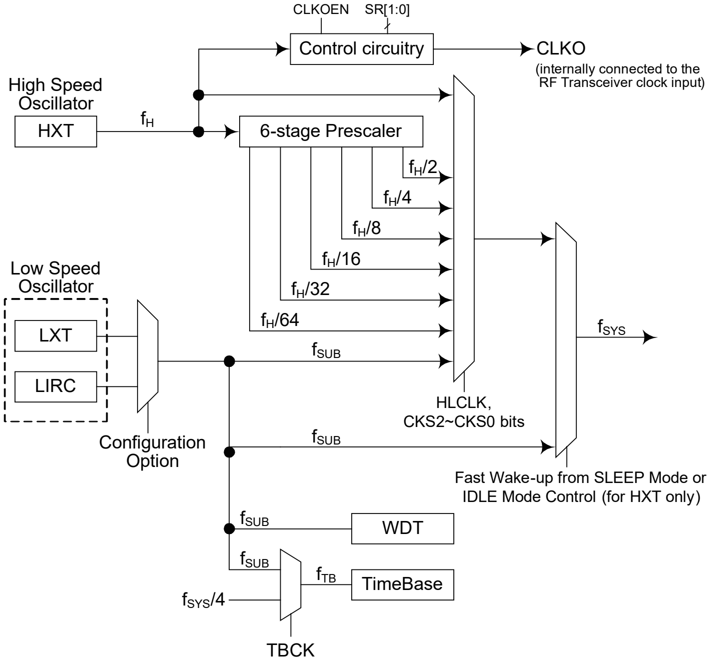
Present day applications require that their microcontrollers have high performance but often still demand that they consume as little power as possible, conflicting requirements that are especially true in battery powered portable applications. The fast clocks required for high performance will by their nature increase current consumption and of course vice-versa, lower speed clocks reduce current consumption. As Holtek has provided these devices with both high and low speed clock sources and the means to switch between them dynamically, the user can optimise the operation of their microcontroller to achieve the best performance/power ratio.

System Clocks

The devices have many different clock sources for both the CPU and peripheral function operation. By providing the user with a wide range of clock options using configuration options and register programming, a clock system can be configured to obtain maximum application performance.

The main system clock can come from a high frequency f_H or low frequency f_{SUB} source and is selected using the HLCLK bit and CKS2~CKS0 bits in the SMOD register. The high speed system clock can be sourced from a HXT oscillator. The low speed system clock source can be sourced from the clock f_{SUB}. If f_{SUB} is selected, then it can be sourced from either the LIRC or LXT oscillator. The other choice, which is a divided version of the high speed system oscillator has a range of $f_H/2$~$f_H/64$. Note that when the system clock source f_{SYS} is switched to f_L from f_H, the high speed oscillation will stop to conserve the power. Thus there is no f_H~$f_H/64$ for peripheral circuit to use.

The f_{SUB} clock is used to provide a substitute clock for the microcontroller just after a wake-up has occurred to enable faster wake-up times. Together with $f_{SYS}/4$ it is also used as one of the clock sources for the Time Base interrupt function. The f_{SUB} clock is also used as a clock source for the Watchdog Timer, TMs or SIM functions.



Operating Mode Clock Configurations

Note: When the system clock source f_{SYS} is switched to f_L from f_H, the high speed oscillation will stop to conserve the power. Thus there is no f_H~$f_H/64$ for peripheral circuit to use.

System Operation Modes

There are six different modes of operation for the microcontroller, each one with its own special characteristics and which can be chosen according to the specific performance and power requirements of the application. There are two modes allowing normal operation of the microcontroller, the NORMAL Mode and SLOW Mode. The remaining four modes, the SLEEP0, SLEEP1, IDLE0 and IDLE1 Mode are used when the microcontroller CPU is switched off to conserve power.

Operating Mode	Description		
	CPU	f_{SYS}	f_{SUB}
NORMAL Mode	On	f_H~f_H/64	On
SLOW Mode	On	f_{SUB}	On
IDLE0 Mode	Off	Off	On
IDLE1 Mode	Off	On	On
SLEEP0 Mode	Off	Off	Off
SLEEP1 Mode	Off	Off	On

NORMAL Mode

As the name suggests this is one of the main operating modes where the microcontroller has all of its functions operational and where the system clock is provided by one of the high speed oscillators. This mode operates allowing the microcontroller to operate normally with a clock source will come from the HXT high speed oscillator. The high speed oscillator will however first be divided by a ratio ranging from 1 to 64, the actual ratio being selected by the CKS2~LCKS0 and HLCLK bits in the SMOD register. Although a high speed oscillator is used, running the microcontroller at a divided clock ratio reduces the operating current.

SLOW Mode

This is also a mode where the microcontroller operates normally although now with a slower speed clock source. The clock source used will be from the low speed oscillators, either the LXT or LIRC oscillators. Running the microcontroller in this mode allows it to run with much lower operating currents. In the SLOW Mode, the f_H is off.

SLEEP0 Mode

The SLEEP Mode is entered when a HALT instruction is executed and the IDLEN bit in the SMOD register is low. In the SLEEP0 mode the CPU will be stopped, and the f_{SUB} clock will be stopped too, and the Watchdog Timer function is disabled. In this mode, the LVDEN is must set to 0. If the LVDEN is set to 1, it won’t enter the SLEEP0 Mode.

SLEEP1 Mode

The SLEEP Mode is entered when a HALT instruction is executed and the IDLEN bit in the SMOD register is low. In the SLEEP1 mode the CPU will be stopped. However, the f_{SUB} clock will continue to operate if the LVDEN is set to 1 or the Watchdog Timer function is enabled as its clock source is from the f_{SUB}.

IDLE0 Mode

The IDLE0 Mode is entered when a HALT instruction is executed and when the IDLEN bit in the SMOD register is high and the FSYSON bit in the CTRL0 register is low. In the IDLE0 Mode the system oscillator will be inhibited from driving the CPU but some peripheral functions will remain operational such as the Watchdog Timer, TMs and SIM. In the IDLE0 Mode, the system oscillator will be stopped. In the IDLE0 Mode the Watchdog Timer clock, f_{SUB}, will be on if the Watchdog Timer function is enabled.

IDLE1 Mode

The IDLE1 Mode is entered when a HALT instruction is executed and when the IDLEN bit in the SMOD register is high and the FSYSON bit in the CTRL0 register is high. In the IDLE1 Mode the system oscillator will be inhibited from driving the CPU but may continue to provide a clock source to keep some peripheral functions operational such as the Watchdog Timer, TMs and SIM. In the IDLE1 Mode, the system oscillator will continue to run, and this system oscillator may be high speed or low speed system oscillator. In the IDLE1 Mode the Watchdog Timer clock, f_{SUB}, will be on if the Watchdog Timer function is enabled.

Control Register

A register, SMOD, together with the FSYSON bit in the CTRL0 register are used for overall control of the internal clocks within the devices.

SMOD Register

Bit	7	6	5	4	3	2	1	0
Name	CKS2	CKS1	CKS0	FSTEN	LTO	HTO	IDLEN	HLCLK
R/W	R/W	R/W	R/W	R/W	R	R	R/W	R/W
POR	1	1	0	0	0	1	1	0

Bit 7~5 **CKS2~CKS0:** The system clock selection when HLCLK is 0

000: f_{SUB} (f_{LIRC} or f_{LXT})
001: f_{SUB} (f_{LIRC} or f_{LXT})
010: f_H/64
011: f_H/32
100: f_H/16
101: f_H/8
110: f_H/4
111: f_H/2

These three bits are used to select which clock is used as the system clock source. In addition to the system clock source, which is the LIRC or LXT clock, a divided version of the high speed system oscillator can also be chosen as the system clock source.

Bit 4 **FSTEN:** Fast Wake-up Control (only for HXT)

0: Disable
1: Enable

This is the Fast Wake-up Control bit which determines if the f_{SUB} clock source is initially used after the devices wake up. When the bit is high, the f_{SUB} clock source can be used as a temporary system clock to provide a faster wake up time as the f_{SUB} clock is available.

Bit 3 **LTO:** Low speed system oscillator ready flag

0: Not ready
1: Ready

This is the low speed system oscillator ready flag which indicates when the low speed system oscillator is stable after power on reset or a wake-up has occurred. The flag will be low when in the SLEEP0 Mode but after a wake-up has occurred, the flag will change to a high level after 1~2 clock cycles if the LIRC oscillator is used or 1024 clock cycles if the LXT oscillator is used.

Bit 2 **HTO:** High speed system oscillator ready flag
0: Not ready
1: Ready
This is the high speed system oscillator ready flag which indicates when the high speed system oscillator is stable. This flag is cleared to 0 by hardware when the devices are powered on and then changes to a high level after the high speed system oscillator is stable. Therefore this flag will always be read as 1 by the application program after device power-on. The flag will be low when in the SLEEP or IDLE0 Mode but after a wake-up has occurred, the flag will change to a high level after 1024 clock cycles as the HXT oscillator is used.

Bit 1 **IDLEN:** IDLE Mode control
0: Disable
1: Enable
This is the IDLE Mode Control bit and determines what happens when the HALT instruction is executed. If this bit is high, when a HALT instruction is executed the devices will enter the IDLE Mode. In the IDLE1 Mode the CPU will stop running but the system clock will continue to keep the peripheral functions operational as the FSYSON bit is high. If FSYSON bit is low, the CPU and the system clock will all stop in IDLE0 mode. If the bit is low the devices will enter the SLEEP Mode when a HALT instruction is executed.

Bit 0 **HLCLK:** System clock selection
0: $f_H/2 \sim f_H/64$ or f_{SUB}
1: f_H
This bit is used to select if the f_H clock or the $f_H/2 \sim f_H/64$ or f_{SUB} clock is used as the system clock. When the bit is high the f_H clock will be selected and if low the $f_H/2 \sim f_H/64$ or f_{SUB} clock will be selected. When system clock switches from the f_H clock to the f_{SUB} clock, the f_H clock will be automatically switched off to conserve power.

CTRL0 Register

Bit	7	6	5	4	3	2	1	0
Name	FSYSON	—	—	—	—	LVRF	LRF	WRF
R/W	R/W	—	—	—	—	R/W	R/W	R/W
POR	0	—	—	—	—	x	0	0

"x": unknown

Bit 7 **FSYSON:** f_{SYS} Control in IDLE Mode
0: Disable
1: Enable
This is the Fast Wake-up Control bit which determines if the f_{SUB} clock source is initially used after the devices wake up.

Bit 6~3 Unimplemented, read as “0”

Bit 2 **LVRF:** LVR function reset flag
0: Not occurred
1: Occurred
This bit is set to 1 when a specific Low Voltage Reset situation occurs. This bit can only be cleared to 0 by the application program.

Bit 1 **LRF:** LVR Control register software reset flag
0: Not occurred
1: Occurred
This bit is set to 1 if the LVRC register contains any non defined LVR voltage register values. This in effect acts like a software-reset function. This bit can only be cleared to 0 by the application program.

Bit 0 **WRF:** WDT Control register software reset flag
0: Not occurred
1: Occurred
This bit is set to 1 by the WDT Control register software reset and cleared by the application program. Note that this bit can only be cleared to 0 by the application program.

The CLKO clock signal is derived from the external crystal oscillator and internally connected to the RF Transceiver clock input. It is recommended to enable the CLKO clock always when the RF transceiver is used.

CTRL1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	CLKOEN	—	—	SR1	SR0
R/W	—	—	—	R/W	—	—	R/W	R/W
POR	—	—	—	0	—	—	0	0

Bit 7~5 Unimplemented, read as "0"

Bit 4 **CLKOEN:** CLKO Output Enable Control
0: Disable
1: Enable
This bit is used to enable or disable the CLKO output signal which is internally connected to the RF transceiver clock input. It is recommended to enable the CLKO output when the RF transceiver is used even if the devices enter the power down mode.

Bit 3~2 Unimplemented, read as "0"

Bit 1~0 **SR1~SR0:** CLKO Output Slew Rate Control

SR[1:0]	V_{CLKO}=3V	V_{CLKO}=5V
00	0.22V/ns	0.5V/ns
01	0.20V/ns	0.4V/ns
10	0.15V/ns	0.3V/ns
11	0.10V/ns	0.2V/ns

These bits are used to control the CLKO output slew rate.

Fast Wake-up

To minimise power consumption the devices can enter the SLEEP or IDLE0 Mode, where the system clock source to the devices will be stopped. However when the devices are woken up again, it can take a considerable time for the original system oscillator to restart, stabilise and allow normal operation to resume. To ensure the devices are up and running as fast as possible a Fast Wake-up function is provided, which allows f_{SUB}, namely the LIRC or LXT oscillator, to act as a temporary clock to first drive the system until the original system oscillator has stabilised. As the clock source for the Fast Wake-up function is f_{SUB}, the Fast Wake-up function is only available in the SLEEP1 and IDLE0 modes. When the devices are woken up from the SLEEP0 mode, the Fast Wake-up function has no effect because the f_{SUB} clock is stopped. The Fast Wake-up enable/disable function is controlled using the FSTEN bit in the SMOD register.

If the HXT oscillator is selected as the NORMAL Mode system clock and the Fast Wake-up function is enabled, then it will take one to two t_{SUB} clock cycles of the LIRC or LXT oscillator for the system to wake-up. The system will then initially run under the f_{SUB} clock source until 1024 HXT clock cycles have elapsed, at which point the HTO flag will switch high and the system will switch over to operating from the HXT oscillator.

If the LXT or LIRC oscillator is used as the system oscillator then it will take 1024 clock cycles of the LXT oscillator or 1~2 cycles of the LIRC oscillator to wake up the system from the SLEEP or IDLE0 Mode. The Fast Wake-up bit, FSTEN will have no effect in these cases.

System Oscillator	FSTEN Bit	Wake-up Time (SLEEP0 Mode)	Wake-up Time (SLEEP1 Mode)	Wake-up Time (IDLE0 Mode)	Wake-up Time (IDLE1 Mode)
HXT	0	1024 HXT cycles	1024 HXT cycles		1~2 HXT cycles
	1	1024 HXT cycles	1~2 f_{SUB} cycles (System runs first with f_{SUB} for 1024 HXT cycles and then switches over to run with the HXT clock)		1~2 HXT cycles
LIRC	x	1~2 LIRC cycles	1~2 LIRC cycles		1~2 LIRC cycles
LXT	x	1024 LXT cycles	1024 LXT cycles		1~2 LXT cycles

"x": don't care

Wake-up Times

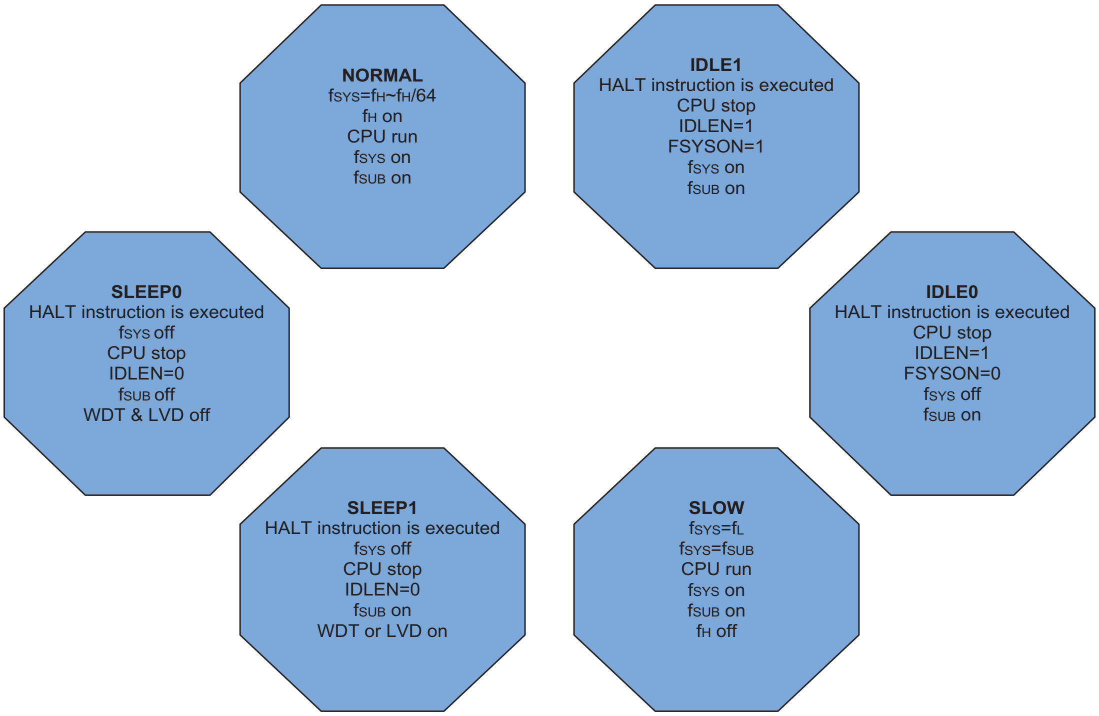
Note that if the Watchdog Timer is disabled, which means that the LIRC or LXT oscillator is off, then there will be no Fast Wake-up function available when the devices wake-up from the SLEEP0 Mode.

Operating Mode Switching

The devices can switch between operating modes dynamically allowing the user to select the best performance/power ratio for the present task in hand. In this way microcontroller operations that do not require high performance can be executed using slower clocks thus requiring less operating current and prolonging battery life in portable applications.

In simple terms, Mode Switching between the NORMAL Mode and SLOW Mode is executed using the HLCLK bit and CKS2~CKS0 bits in the SMOD register while Mode Switching from the NORMAL/SLOW Modes to the SLEEP/IDLE Modes is executed via the HALT instruction. When a HALT instruction is executed, whether the devices enter the IDLE Mode or the SLEEP Mode is determined by the condition of the IDLEN bit in the SMOD register and FSYSON in the WDTC register.

When the HLCLK bit switches to a low level, which implies that clock source is switched from the high speed clock source, f_H, to the clock source, $f_H/2$~$f_H/64$ or f_L. If the clock is from the f_L, the high speed clock source will stop running to conserve power. When this happens it must be noted that the $f_H/16$ and $f_H/64$ internal clock sources will also stop running, which may affect the operation of other internal functions such as the TMs and the SIM. The accompanying flowchart shows what happens when the devices move between the various operating modes.



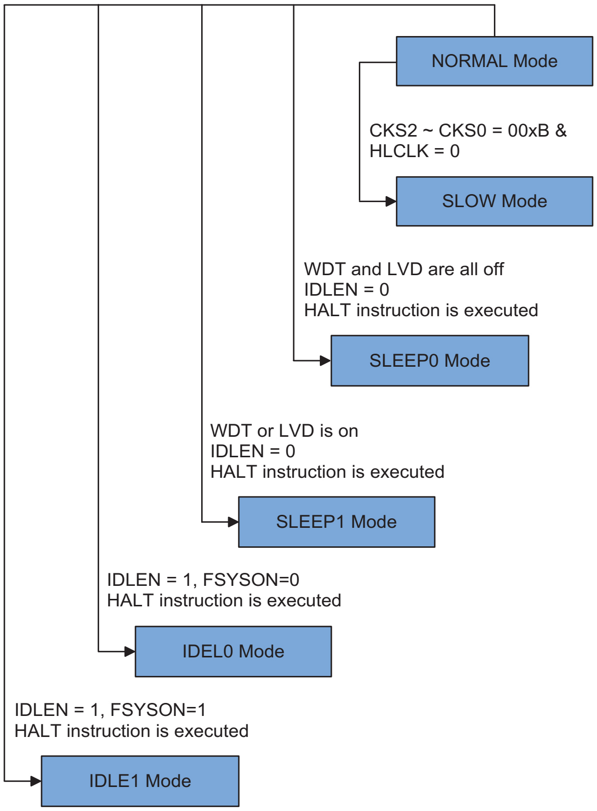
NORMAL Mode to SLOW Mode Switching

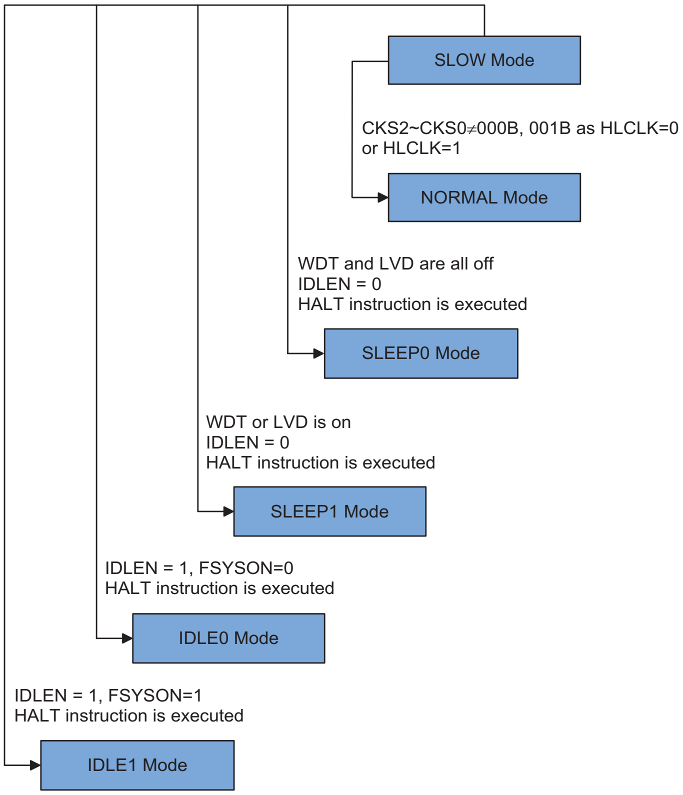
When running in the NORMAL Mode, which uses the high speed system oscillator, and therefore consumes more power, the system clock can switch to run in the SLOW Mode by set the HLCLK bit to 0 and set the CKS2~CKS0 bits to 000B or 001B in the SMOD register. This will then use the low speed system oscillator which will consume less power. Users may decide to do this for certain operations which do not require high performance and can subsequently reduce power consumption.

The SLOW Mode is sourced from the LIRC or LXT oscillator and therefore requires these oscillators to be stable before full mode switching occurs. This is monitored using the LTO bit in the SMOD register.

SLOW Mode to NORMAL Mode Switching

In SLOW Mode the system uses the LIRC or LXT low speed system oscillator. To switch back to the NORMAL Mode, where the high speed system oscillator is used, the HLCLK bit should be set to 1 or HLCLK bit is 0 but CKS2~CKS0 is set to 010B, 011B, 100B, 101B, 110B or 111B. As a certain amount of time will be required for the high frequency clock to stabilise, the status of the HTO bit is checked. The amount of time required for high speed system oscillator stabilization depends upon which high speed system oscillator type is used.





Entering the SLEEP0 Mode

There is only one way for the devices to enter the SLEEP0 Mode and that is to execute the HALT instruction in the application program with the IDLEN bit in SMOD register equal to 0 and the WDT and LVD both off. When this instruction is executed under the conditions described above, the following will occur:

- The system clock, WDT clock and Time Base clock will be stopped and the application program will stop at the HALT instruction.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and stopped.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Entering the SLEEP1 Mode

There is only one way for the devices to enter the SLEEP1 Mode and that is to execute the HALT instruction in the application program with the IDLEN bit in SMOD register equal to 0 and the WDT or LVD on. When this instruction is executed under the conditions described above, the following will occur:

- The system clock and Time Base clock will be stopped and the application program will stop at the HALT instruction, but the WDT or LVD will remain with the clock source coming from the f_{SUB} clock.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and resume counting as the WDT function is enabled.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Entering the IDLE0 Mode

There is only one way for the devices to enter the IDLE0 Mode and that is to execute the HALT instruction in the application program with the IDLEN bit in SMOD register equal to 1 and the FSYSON bit in the CTRL0 register equal to 0. When this instruction is executed under the conditions described above, the following will occur:

- The system clock will be stopped and the application program will stop at the HALT instruction, but the Time Base clock and f_{SUB} clock will be on.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and resume counting as the WDT is enabled.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Entering the IDLE1 Mode

There is only one way for the devices to enter the IDLE1 Mode and that is to execute the HALT instruction in the application program with the IDLEN bit in SMOD register equal to 1 and the FSYSON bit in the CTRL0 register equal to 1. When this instruction is executed under the conditions described above, the following will occur:

- The system clock, Time Base clock and f_{SUB} clock will be on and the application program will stop at the HALT instruction.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and resume counting as the WDT is enabled.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Standby Current Considerations

As the main reason for entering the SLEEP or IDLE Mode is to keep the current consumption of the devices to as low a value as possible, perhaps only in the order of several micro-amps except in the IDLE1 Mode, there are other considerations which must also be taken into account by the circuit designer if the power consumption is to be minimised. Special attention must be made to the I/O pins on the devices. All high-impedance input pins must be connected to either a fixed high or low level as any floating input pins could create internal oscillations and result in increased current consumption. This also applies to devices which have different package types, as there may be unbonbed pins. These must either be setup as outputs or if setup as inputs must have pull-high resistors connected.

Care must also be taken with the loads, which are connected to I/O pins, which are setup as outputs. These should be placed in a condition in which minimum current is drawn or connected only to external circuits that do not draw current, such as other CMOS inputs. Also note that additional standby current will also be required if the configuration options have enabled the LIRC oscillator.

In the IDLE1 Mode the system oscillator is on, if the system oscillator is from the high speed system oscillator, the additional standby current will also be perhaps in the order of several hundred micro-amps.

Wake-up

After the system enters the SLEEP or IDLE Mode, it can be woken up from one of various sources listed as follows:

- An external reset
- An external falling edge on Port A
- A system interrupt
- A WDT overflow

If the system is woken up by an external reset, the devices will experience a full system reset, however, if the devices are woken up by a WDT overflow, a Watchdog Timer reset will be initiated. Although both of these wake-up methods will initiate a reset operation, the actual source of the wake-up can be determined by examining the TO and PDF flags. The PDF flag is cleared by a system power-up or executing the clear Watchdog Timer instructions and is set when executing the HALT instruction. The TO flag is set if a WDT time-out occurs, and causes a wake-up that only resets the Program Counter and Stack Pointer, the other flags remain in their original status.

Each pin on Port A can be setup using the PAWU register to permit a negative transition on the pin to wake-up the system. When a Port A pin wake-up occurs, the program will resume execution at the instruction following the HALT instruction. If the system is woken up by an interrupt, then two possible situations may occur. The first is where the related interrupt is disabled or the interrupt is enabled but the stack is full, in which case the program will resume execution at the instruction following the HALT instruction. In this situation, the interrupt which woke-up the devices will not be immediately serviced, but will rather be serviced later when the related interrupt is finally enabled or when a stack level becomes free. The other situation is where the related interrupt is enabled and the stack is not full, in which case the regular interrupt response takes place. If an interrupt request flag is set high before entering the SLEEP or IDLE Mode, the wake-up function of the related interrupt will be disabled.

Programming Considerations

The high speed and low speed oscillators both use the same SST counter. For example, if the system is woken up from the SLEEP0 Mode and both the HXT and LXT oscillators need to start-up from an off state. The LXT oscillator uses the SST counter after HXT oscillator has finished its SST period.

- If the devices are woken up from the SLEEP0 Mode to the NORMAL Mode, the high speed system oscillator needs an SST period. The devices will execute first instruction after HTO is "1". At this time, the LXT oscillator may not be stable if f_{SUB} is from LXT oscillator. The same situation occurs in the power-on state. The LXT oscillator is not ready yet when the first instruction is executed.
- If the devices are woken up from the SLEEP1 Mode to NORMAL Mode, and the system clock source is from the HXT oscillator and FSTEN is 1, the system clock can first be switched to the f_{SUB} clock after wake up.
- There are peripheral functions, such as WDT and TMs, for which the f_{SYS} is used. If the system clock source is switched from f_H to f_{SUB}, the clock source to the peripheral functions mentioned above will change accordingly.
- The on/off condition of f_{SUB} depends upon whether the WDT or LVD function is enabled or disabled as the WDT clock source is selected from f_{SUB}.

Watchdog Timer

The Watchdog Timer is provided to prevent program malfunctions or sequences from jumping to unknown locations, due to certain uncontrollable external events such as electrical noise.

Watchdog Timer Clock Source

The Watchdog Timer clock source is provided by the internal clock, f_{SUB}. The f_{SUB} clock can be sourced from either the LIRC or LXT oscillator selected by a configuration option. The LIRC internal oscillator has an approximate period of 32kHz at a supply voltage of 5V. However, it should be noted that this specified internal clock period can vary with V_{DD}, temperature and process variations. The LXT oscillator us supplied by an external 32.768kHz crystal. The Watchdog Timer source clock is then subdivided by a ratio of 2^8 to 2^{18} to give longer timeouts, the actual value being chosen using the WS2~WS0 bits in the WDTC register.

Watchdog Timer Control Register

A single register, WDTC, controls the required timeout period as well as the enable/disable operation. This register together with several configuration options control the overall operation of the Watchdog Timer.

WDTC Register

Bit	7	6	5	4	3	2	1	0
Name	WE4	WE3	WE2	WE1	WE0	WS2	WS1	WS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	1	0	1	0	0	1	1

Bit 7~3 **WE4~WE0:** WDT Function Enable control
10101: Disabled
01010: Enable
Other Value: Reset MCU

If these bits are changed due to adverse environmental conditions, the microcontroller will be reset. The reset operation will be activated after 2~3 f_{SUB} clock cycles and the WRF bit in the CTRL0 register will be set to 1

Bit 2~0 **WS2~WS0:** WDT Time-out period selection
000: $2^8/f_S$
001: $2^{10}/f_S$
010: $2^{12}/f_S$
011: $2^{14}/f_S$
100: $2^{15}/f_S$
101: $2^{16}/f_S$
110: $2^{17}/f_S$
111: $2^{18}/f_S$

These three bits determine the division ratio of the Watchdog Timer source clock, which in turn determines the timeout period.

CTRL0 Register

Bit	7	6	5	4	3	2	1	0
Name	FSYSON	—	—	—	—	LVRF	LRF	WRF
R/W	R/W	—	—	—	—	R/W	R/W	R/W
POR	0	—	—	—	—	x	0	0

"x": unknown

Bit 7 **FSYSON:** f_{SYS} Control in IDLE Mode
Described elsewhere

Bit 6~3 Unimplemented, read as "0"

Bit 2 **LVRF:** LVR function reset flag
Described elsewhere

Bit 1 **LRF:** LVR Control register software reset flag
Described elsewhere

Bit 0 **WRF:** WDT Control register software reset flag
0: Not occurred
1: Occurred

This bit is set to 1 by the WDT Control register software reset and cleared by the application program. Note that this bit can only be cleared to 0 by the application program.

Watchdog Timer Operation

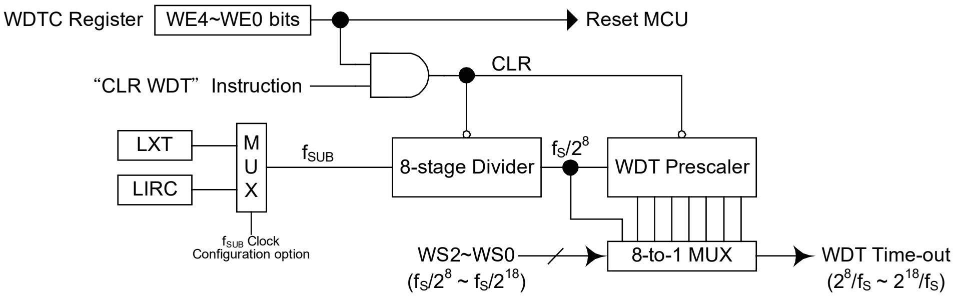
The Watchdog Timer operates by providing a device reset when its timer overflows. This means that in the application program and during normal operation the user has to strategically clear the Watchdog Timer before it overflows to prevent the Watchdog Timer from executing a reset. This is done using the clear watchdog instructions. If the program malfunctions for whatever reason, jumps to an unknown location, or enters an endless loop, these clear instructions will not be executed in the correct manner, in which case the Watchdog Timer will overflow and reset the device. With regard to the Watchdog Timer enable/disable function, there are five bits, WE4~WE0, in the WDTC register to offer the enable/disable control and reset control of the Watchdog Timer. The WDT function will be disabled when the WE4~WE0 bits are set to a value of 10101B while the WDT function will be enabled if the WE4~WE0 bits are equal to 01010B. If the WE4~WE0 bits are set to any other values, other than 01010B and 10101B, it will reset the device after 2~3 f_{SUB} clock cycles. After power on these bits will have a value of 01010B.

WDT Function Control	WE4~WE0 Bits	WDT Function
Application Program Enabled	10101B	Disable
	01010B	Enable
	Any other value	Reset MCU

Watchdog Timer Enable/Disable Control

Under normal program operation, a Watchdog Timer time-out will initialise a device reset and set the status bit TO. However, if the system is in the SLEEP or IDLE Mode, when a Watchdog Timer time-out occurs, the TO bit in the status register will be set and only the Program Counter and Stack Pointer will be reset. Three methods can be adopted to clear the contents of the Watchdog Timer. The first is a WDT reset, which means a certain value except 01010B and 10101B written into the WE4~WE0 bit filed, the second is using the Watchdog Timer software clear instructions and the third is via a HALT instruction.

There is only one method of using software instruction to clear the Watchdog Timer. That is to use the single "CLR WDT" instruction to clear the WDT.



Watchdog Timer

Reset and Initialisation

A reset function is a fundamental part of any microcontroller ensuring that the device can be set to some predetermined condition irrespective of outside parameters. The most important reset condition is after power is first applied to the microcontroller. In this case, internal circuitry will ensure that the microcontroller, after a short delay, will be in a well defined state and ready to execute the first program instruction. After this power-on reset, certain important internal registers will be set to defined states before the program commences. One of these registers is the Program Counter, which will be reset to zero forcing the microcontroller to begin program execution from the lowest Program Memory address.

In addition to the power-on reset, situations may arise where it is necessary to forcefully apply a reset condition when the microcontroller is running. One example of this is where after power has been applied and the microcontroller is already running, the $\overline{RES}$ line is forcefully pulled low. In such a case, known as a normal operation reset, some of the microcontroller registers remain unchanged allowing the microcontroller to precede with normal operation after the reset line is allowed to return high.

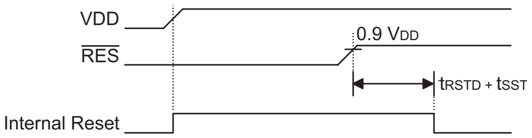
Another type of reset is when the Watchdog Timer overflows and resets the microcontroller. All types of reset operations result in different register conditions being setup. Another reset exists in the form of a Low Voltage Reset, LVR, where a full reset, similar to the $\overline{RES}$ reset is implemented in situations where the power supply voltage falls below a certain threshold.

Reset Functions

There are five ways in which a microcontroller reset can occur, through events occurring both internally and externally:

Power-on Reset

The most fundamental and unavoidable reset is the one that occurs after power is first applied to the microcontroller. As well as ensuring that the Program Memory begins execution from the first memory address, a power-on reset also ensures that certain other registers are preset to known conditions. All the I/O port and port control registers will power up in a high condition ensuring that all pins will be first set to inputs.



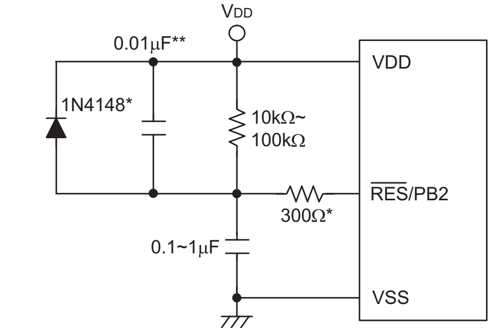
Note: t_{RSTD} is power-on delay, typical time=50ms

Power-on Reset Timing Chart

$\overline{RES}$ Pin Reset

As the reset pin is shared with PB2, the reset function must be selected using a configuration option. Although the microcontroller has an internal RC reset function, if the VDD power supply rise time is not fast enough or does not stabilise quickly at power-on, the internal reset function may be incapable of providing proper reset operation. For this reason it is recommended that an external RC network is connected to the $\overline{RES}$ pin, whose additional time delay will ensure that the $\overline{RES}$ pin remains low for an extended period to allow the power supply to stabilise. During this time delay, normal operation of the microcontroller will be inhibited. After the $\overline{RES}$ line reaches a certain voltage value, the reset delay time t_{RSTD} is invoked to provide an extra delay time after which the microcontroller will begin normal operation. The abbreviation SST in the figures stands for System

Start-up Timer. For most applications a resistor connected between VDD and the $\overline{\text{RES}}$ pin and a capacitor connected between VSS and the $\overline{\text{RES}}$ pin will provide a suitable external reset circuit. Any wiring connected to the $\overline{\text{RES}}$ pin should be kept as short as possible to minimise any stray noise interference. For applications that operate within an environment where more noise is present the Enhanced Reset Circuit shown is recommended.

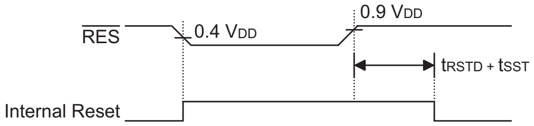


Note: "*" It is recommended that this component is added ESD protection.
"**" It is recommended that this component is added in environments where power line noise is significant.

External $\overline{\text{RES}}$ Circuit

More information regarding external reset circuits is located in Application Note HA0075E on the Holtek website.

Pulling the $\overline{\text{RES}}$ Pin low using external hardware will also execute a device reset. In this case, as in the case of other resets, the Program Counter will reset to zero and program execution initiated from this point.



Note: t_{RSTD} is power-on delay, typical time=16.7ms

$\overline{\text{RES}}$ Reset Timing Chart

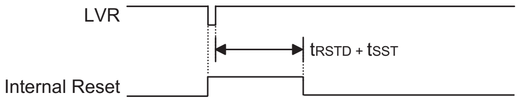
When the reset pin is driven low by external hardware, most of the microcontroller pins will be forced into a high impedance condition. However special attention must be made to the PA2/CX/AN2 as the pin will be forced into a logical output low condition when the reset pin is held low. For this reason it is recommended that the pin is not connected to low impedance source in the application circuit to eliminate the possibility of the low impedance being connected together. This situation only occurs when the reset pin is pulled low by external hardware and not during a power on or other reset type.

Pin Name	Pin Status
PA2/CX/AN2	Output Low
Other pins	High Impedance

Reset Pin Forced Low – Pin Status

Low Voltage Reset – LVR

The microcontroller contains a low voltage reset circuit in order to monitor the supply voltage of the device. The LVR function is always enabled with a specific LVR voltage, V_{LVR}. If the supply voltage of the device drops to within a range of 0.9V~V_{LVR} such as might occur when changing the battery, the LVR will automatically reset the device internally and the LVRF bit in the CTRL0 register will also be set to 1. For a valid LVR signal, a low supply voltage, i.e., a voltage in the range between 0.9V~V_{LVR} must exist for a time greater than that specified by t_{LVR} in the A.C. characteristics. If the low supply voltage state does not exceed this value, the LVR will ignore the low supply voltage and will not perform a reset function. The actual V_{LVR} value can be selected by the LVS bits in the LVRC register. If the LVS7~LVS0 bits have any other value, which may perhaps occur due to adverse environmental conditions such as noise, the LVR will reset the device after 2~3 f_{SUB} clock cycles. When this happens, the LRF bit in the CTRL0 register will be set to 1. After power on the register will have the value of 01010101B. Note that the LVR function will be automatically disabled when the device enters the power down mode.



Note: t_{RSTD} is power-on delay, typical time=16.7ms

Low Voltage Reset Timing Chart

- LVRC Register

Bit	7	6	5	4	3	2	1	0
Name	LVS7	LVS6	LVS5	LVS4	LVS3	LVS2	LVS1	LVS0
R/W	R/W	R/W	R/W	R/W	R/w	R/w	R/W	R/W
POR	0	1	0	1	0	1	0	1

Bit 7~0 **LVS7~LVS0:** LVR voltage select
01010101: 2.1V
00110011: 2.55V
10011001: 3.15V
10101010: 3.8V
Any other values: Generates MCU reset – register is reset to POR value

When an actual low voltage condition occurs, as specified by one of the four defined LVR voltage values above, an MCU reset will be generated. The reset operation will be activated after 2~3 f_{SUB} clock cycles. In this situation the register contents will remain the same after such a reset occurs.

Any register value, other than the four defined register values above, will also result in the generation of an MCU reset. The reset operation will be activated after 2~3 f_{SUB} clock cycles. However in this situation the register contents will be reset to the POR value.

- CTRL0 Register

Bit	7	6	5	4	3	2	1	0
Name	FSYSON	—	—	—	—	LVRF	LRF	WRF
R/W	R/W	—	—	—	—	R/W	R/W	R/W
POR	0	—	—	—	—	x	0	0

"x": unknown

Bit 7 **FSYSON:** f_{SYS} Control in IDLE Mode
Described elsewhere.

Bit 6~3 Unimplemented, read as "0"

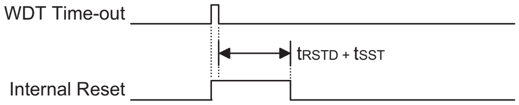
Bit 2 **LVRF:** LVR function reset flag
0: Not occurred
1: occurred
This bit is set to 1 when a specific Low Voltage Reset situation condition occurs. This bit can only be cleared to 0 by the application program.

Bit 1 **LRF:** LVR Control register software reset flag
0: Not occurred
1: occurred
This bit is set to 1 if the LVRC register contains any non defined LVR voltage register values. This in effect acts like a software-reset function. This bit can only be cleared to 0 by the application program.

bit 0 **WRF:** WDT Control register software reset flag
Described elsewhere.

Watchdog Time-out Reset during Normal Operation

The Watchdog time-out Reset during normal operation is the same as a hardware $\overline{RES}$ pin reset except that the Watchdog time-out flag TO will be set to "1".

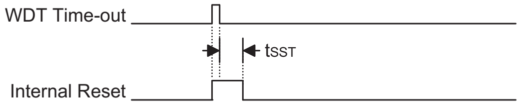


Note: t_{RSTD} is power-on delay, typical time=16.7ms

WDT Time-out Reset during Normal Operation Timing Chart

Watchdog Time-out Reset during SLEEP or IDLE Mode

The Watchdog time-out Reset during SLEEP or IDLE Mode is a little different from other kinds of reset. Most of the conditions remain unchanged except that the Program Counter and the Stack Pointer will be cleared to "0" and the TO flag will be set to "1". Refer to the A.C. Characteristics for t_{SST} details.



Note: The SST is 1024 clock cycles for HXT and LXT. The SST is 1~2 clock cycles for LIRC.

WDT Time-out Reset during SLEEP or IDLE Timing Chart

Reset Initial Conditions

The different types of reset described affect the reset flags in different ways. These flags, known as PDF and TO are located in the status register and are controlled by various microcontroller operations, such as the SLEEP or IDLE Mode function or Watchdog Timer. The reset flags are shown in the table:

TO	PDF	RESET Conditions
0	0	Power-on Reset
u	u	$\overline{RES}$ or LVR reset during NORMAL or SLOW Mode operation
1	u	WDT time-out reset during NORMAL or SLOW Mode operation
1	1	WDT time-out reset during IDLE or SLEEP Mode operation

Note: "u" stands for unchanged.

The following table indicates the way in which the various components of the microcontroller are affected after a power-on reset occurs.

Item	Condition After RESET
Program Counter	Reset to zero
Interrupts	All interrupt will be disabled
WDT	Clear after reset, WDT begins counting
Timer/Event Counter	Timer Counter will be turned off
Input/Output Ports	I/O ports will be setup as inputs and ANn as A/D input pins
Stack Pointer	Stack Pointer will point to the top of the stack

The different kinds of resets all affect the internal registers of the microcontroller in different ways. To ensure reliable continuation of normal program execution after a reset occurs, it is important to know what condition the microcontroller is in after a particular reset occurs. The following table describes how each type of reset affects each of the microcontroller internal registers.

Register	BC66F840	BC66F850	BC66F860	Power-on Reset	$\overline{RES}$ or LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
IAR0	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP0	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
IAR1	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	●	●	●	0000 0000	uuuu uuuu	uuuu uuuu	uuuu uuuu
BP	●			---- ---0	---- ---0	---- ---0	---- ---u
BP		●		---- --00	---- --00	---- --00	---- --uu
BP			●	--0- --00	--0- --00	--0- --00	--u- --uu
ACC	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBHP	●			---- xxxx	---- uuuu	---- uuuu	---- uuuu
TBHP		●		---x xxxx	---u uuuu	---u uuuu	---u uuuu
TBHP			●	--xx xxxx	--uu uuuu	--uu uuuu	--uu uuuu
STATUS	●	●	●	--00 xxxx	--uu uuuu	--1u uuuu	--11 uuuu
SMOD	●	●	●	1100 0110	1100 x110	1100 x110	uuuu uuuu

Register	BC66F840	BC66F850	BC66F860	Power-on Reset	$\overline{\text{RES}}$ or LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
LVDC	●	●	●	--00 -000	--00 -000	--00 -000	--uu -uuu
INTEG	●	●	●	---- 0000	---- 0000	---- 0000	---- uuuu
INTC0	●	●	●	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	●			000- 000-	000- 000-	000- 000-	uuu- uuu-
INTC1		●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC2	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI0	●	●	●	--00 --00	--00 --00	--00 --00	--uu --uu
MFI1		●	●	--00 --00	--00 --00	--00 --00	--uu --uu
MFI2	●	●	●	--00 --00	--00 --00	--00 --00	--uu --uu
MFI3	●	●	●	-000 -000	-000 -000	-000 -000	-uuu -uuu
MFI4	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAWU	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAPU	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PA	●	●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	●	●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBPU	●			--00 0000	--00 0000	--00 0000	--uu uuuu
PB	●			--11 1111	--11 1111	--11 1111	--uu uuuu
PBC	●			--11 1111	--11 1111	--11 1111	--uu uuuu
PBPU		●	●	-000 0000	-000 0000	-000 0000	-uuu uuuu
PB		●	●	-111 1111	-111 1111	-111 1111	-uuu uuuu
PBC		●	●	-111 1111	-111 1111	-111 1111	-uuu uuuu
PCPU	●			-000 0000	-000 0000	-000 0000	-uuu uuuu
PC	●			-111 1111	-111 1111	-111 1111	-uuu uuuu
PCC	●			-111 1111	-111 1111	-111 1111	-uuu uuuu
PCPU		●		--00 0000	--00 0000	--00 0000	--uu uuuu
PC		●		--11 1111	--11 1111	--11 1111	--uu uuuu
PCC		●		--11 1111	--11 1111	--11 1111	--uu uuuu
PCPU			●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PC			●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCC			●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDPU	●	●	●	--00 0000	--00 0000	--00 0000	--uu uuuu
PD	●	●	●	--11 1111	--11 1111	--11 1111	--uu uuuu
PDC	●	●	●	--11 1111	--11 1111	--11 1111	--uu uuuu
PEPU		●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PE		●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEC		●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFPU			●	---- 0000	---- 0000	---- 0000	---- uuuu
PF			●	---- 1111	---- 1111	---- 1111	---- uuuu
PFC			●	---- 1111	---- 1111	---- 1111	---- uuuu
TMPC	●			---0 00-0	---0 00-0	---0 00-0	---u uu-u

Register	BC66F840	BC66F850	BC66F860	Power-on Reset	$\overline{\text{RES}}$ or LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
TMPC		●	●	---0 0000	---0 0000	---0 0000	---u uuuu
WDTC	●	●	●	0101 0011	0101 0011	0101 0011	uuuu uuuu
TBC	●	●	●	0011 0111	0011 0111	0011 0111	uuuu uuuu
EEA	●			-000 0000	-000 0000	-000 0000	-uuu uuuu
EEA		●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
EED	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
EEC	●	●	●	---- 0000	---- 0000	---- 0000	---- uuuu
ADRL (ADRFS=0)	●	●	●	xxxx ----	xxxx ----	xxxx ----	uuuu ----
ADRL (ADRFS=1)	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADRH (ADRFS=0)	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADRH (ADRFS=1)	●	●	●	---- xxxx	---- xxxx	---- xxxx	---- uuuu
ADCR0	●			0110 -000	0110 -000	0110 -000	uuuu -uuu
ADCR0		●	●	0110 0000	0110 0000	0110 0000	uuuu uuuu
ADCR1	●	●	●	00-0 -000	00-0 -000	00-0 -000	uu-u -uuu
ACERL	●	●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
ACERH		●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
CPC	●	●	●	1000 0--1	1000 0--1	1000 0--1	uuuu u--u
CTRL0	●	●	●	0--- -x00	0--- -000	0--- -000	u--- -uuu
CTRL1	●	●	●	---0 --00	---0 --00	---0 --00	---u --uu
LVRC	●	●	●	0101 0101	uuuu uuuu	0101 0101	uuuu uuuu
TM0C0	●	●	●	0000 0---	0000 0---	0000 0---	uuuu u---
TM0C1	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0DL	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0DH	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0AL	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0AH	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0RP	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1C0		●	●	0000 0---	0000 0---	0000 0---	uuuu u---
TM1C1		●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1DL		●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1DH		●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1AL		●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1AH		●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1RP		●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2C0	●	●	●	0000 0---	0000 0---	0000 0---	uuuu u---
TM2C1	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2DL	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2DH	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	BC66F840	BC66F850	BC66F860	Power-on Reset	$\overline{\text{RES}}$ or LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
TM2AL	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2AH	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM2RP	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3C0	●	●	●	0000 0---	0000 0---	0000 0---	uuuu u---
TM3C1	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3C2	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3DL	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3DH	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3AL	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3AH	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3BL	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3BH	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM3RP	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
SIMC0	●	●	●	1110 000-	1110 000-	1110 000-	uuuu uuu-
SIMC1	●	●	●	1000 0001	1000 0001	1000 0001	uuuu uuuu
SIMD	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SIMC2/SIMA	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
I2CTOC	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
SPIAC0	●	●	●	1110 0000	1110 0000	1110 0000	uuuu uuuu
SPIAC1	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
SPIAD	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu

Note: “u” stands for unchanged
“x” stands for “unknown”
“-“ stands for unimplemented

Input/Output Ports

Holtek microcontrollers offer considerable flexibility on their I/O ports. With the input or output designation of every pin fully under user program control, pull-high selections for all ports and wake-up selections on certain pins, the user is provided with an I/O structure to meet the needs of a wide range of application possibilities.

The devices provide bidirectional input/output lines labeled with port names such as PA and PB, etc. These I/O ports are mapped to the RAM Data Memory with specific addresses as shown in the Special Purpose Data Memory table. All of these I/O ports can be used for input and output operations. For input operation, these ports are non-latching, which means the inputs must be ready at the T2 rising edge of instruction "MOV A, [m]", where m denotes the port address. For output operation, all the data is latched and remains unchanged until the output latch is rewritten.

I/O Resistor Lists

BC66F840

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PAPU	PAPU6	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PBPU	—	—	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PB	—	—	PB5	PB4	PB3	PB2	PB1	PB0
PBC	—	—	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PCPU	—	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PC	—	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PCC	—	PCC6	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PDPU	—	—	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
PD	—	—	PD5	PD4	PD3	PD2	PD1	PD0
PDC	—	—	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0

BC66F850

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PAPU	PAPU6	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PBPU	—	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PB	—	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PBC	—	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PCPU	—	—	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PC	—	—	PC5	PC4	PC3	PC2	PC1	PC0
PCC	—	—	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PDPU	—	—	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
PD	—	—	PD5	PD4	PD3	PD2	PD1	PD0
PDC	—	—	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
PEPU	PEPU6	PEPU6	PEPU5	PEPU4	PEPU3	PEPU2	PEPU1	PEPU0
PE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
PEC	PEC7	PEC6	PEC5	PEC4	PEC3	PEC2	PEC1	PEC0

BC66F860

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PAPU	PAPU6	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PBPU	—	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PB	—	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PBC	—	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PCPU	PCPU7	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PC	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PCC	PCC7	PCC6	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PDPU	—	—	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
PD	—	—	PD5	PD4	PD3	PD2	PD1	PD0
PDC	—	—	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
PEPU	PEPU6	PEPU6	PEPU5	PEPU4	PEPU3	PEPU2	PEPU1	PEPU0
PE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
PEC	PEC7	PEC6	PEC5	PEC4	PEC3	PEC2	PEC1	PEC0
PFPU	—	—	—	—	PFPU3	PFPU2	PFPU1	PFPU0
PF	—	—	—	—	PF3	PF2	PF1	PF0
PFC	—	—	—	—	PFC3	PFC2	PFC1	PFC0

"—": Unimplemented, read as "0"

PAWUn: PA wake-up function control
0: Disable
1: Enable

PAn/PBn/PCn/PDn/PEn/PFn: I/O Data bit
0: Data 0
1: Data 1

PACn/PBCn/PCCn/PDCn/PECn/PFCn: I/O type selection
0: Output
1: Input

PAPUn/PBPUn/PCPUn/PDPUn/PEPUn/PFPUn: Pull-high function control
0: Disable
1: Enable

Pull-high Resistors

Many product applications require pull-high resistors for their switch inputs usually requiring the use of an external resistor. To eliminate the need for these external resistors, all I/O pins, when configured as an input have the capability of being connected to an internal pull-high resistor. These pull-high resistors are selected using registers PAPU~PFPU and are implemented using weak PMOS transistors.

Port A Wake-up

The HALT instruction forces the microcontroller into the SLEEP or IDLE Mode which preserves power, a feature that is important for battery and other low-power applications. Various methods exist to wake-up the microcontroller, one of which is to change the logic condition on one of the Port A pins from high to low. This function is especially suitable for applications that can be woken up via external switches. Each pin on Port A can be selected individually to have this wake-up feature using the PAWU register.

PAWU Register

Bit	7	6	5	4	3	2	1	0
Name	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

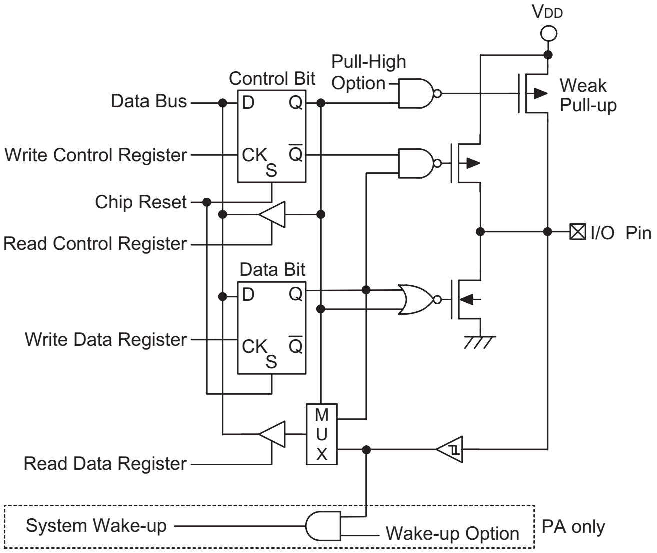
Bit 7~0 **PAWU7~PAWU0:** Port A bit 7~bit 0 Wake-up Control
0: Disable
1: Enable

I/O Port Control Registers

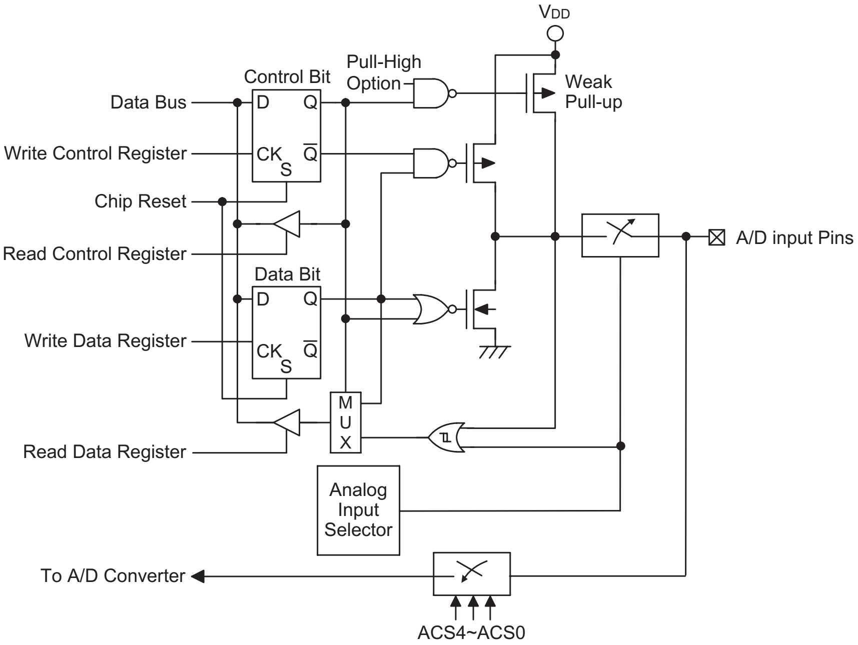
Each I/O port has its own control register known as PAC~PFC, to control the input/output configuration. With this control register, each CMOS output or input can be reconfigured dynamically under software control. Each pin of the I/O ports is directly mapped to a bit in its associated port control register. For the I/O pin to function as an input, the corresponding bit of the control register must be written as 1. This will then allow the logic state of the input pin to be directly read by instructions. When the corresponding bit of the control register is written as 0, the I/O pin will be setup as a CMOS output. If the pin is currently setup as an output, instructions can still be used to read the output register. However, it should be noted that the program will in fact only read the status of the output data latch and not the actual logic status of the output pin.

I/O Pin Structures

The accompanying diagrams illustrate the internal structures of some generic I/O pin types. As the exact logical construction of the I/O pin will differ from these drawings, they are supplied as a guide only to assist with the functional understanding of the I/O pins. The wide range of pin-shared structures does not permit all types to be shown.



Generic Input/Output Structure



A/D Input/Output Structure

Programming Considerations

Within the user program, one of the first things to consider is port initialisation. After a reset, all of the I/O data and port control registers will be set high. This means that all I/O pins will default to an input state, the level of which depends on the other connected circuitry and whether pull-high selections have been chosen. If the port control registers are then programmed to setup some pins as outputs, these output pins will have an initial high output value unless the associated port data registers are first programmed. Selecting which pins are inputs Within the user program, one of the first things to consider is port initialisation. After a reset, all of the I/O data and port control registers will be set high. This means that all I/O pins will default to an input state, the level of which depends on the other connected circuitry and whether pull-high selections have been chosen. If the port control registers, PAC and PBC, are then programmed to setup some pins as outputs, these output pins will have an initial high output value unless the associated port data registers, PA and PB, are first programmed. Selecting which pins are inputs and which are outputs can be achieved byte-wide by loading the correct values into the appropriate port control register or by programming individual bits in the port control register using the SET [m].i and CLR [m].i instructions. Note that when using these bit control instructions, a read-modify-write operation takes place. The microcontroller must first read in the data on the entire port, modify it to the required new bit values and then rewrite this data back to the output ports.

The power-on reset condition of the A/D converter control registers ensures that any A/D input pins – which are always shared with other I/O functions – will be setup as analog inputs after a reset. Although these pins will be configured as A/D inputs after a reset, the A/D converter will not be switched on. It is therefore important to note that if it is required to use these pins as I/O digital input pins or as other functions, the A/D converter control registers must be correctly programmed to remove the A/D function. Note also that as the A/D channel is enabled, any internal pull-high resistor connections will be removed.

Port A has the additional capability of providing wake-up functions. When the devices are in the SLEEP or IDLE Mode, various methods are available to wake the devices up. One of these is a high to low transition of any of the Port A pins. Single or multiple pins on Port A can be setup to have this function.

Timer Modules – TM

One of the most fundamental functions in any microcontroller devices is the ability to control and measure time. To implement time related functions each device includes several Timer Modules, abbreviated to the name TM. The TMs are multi-purpose timing units and serve to provide operations such as Timer/Counter, Input Capture, Compare Match Output and Single Pulse Output as well as being the functional unit for the generation of PWM signals. Each of the TMs has either two individual interrupts. The addition of input and output pins for each TM ensures that users are provided with timing units with a wide and flexible range of features.

The common features of the different TM types are described here with more detailed information provided in the individual Compact, Standard and Enhanced TM sections.

Introduction

The devices contain up to four TMs with each TM having a reference name of TM0, TM1, TM2 and TM3. Each individual TM can be categorised as a certain type, namely Compact Type TM (CTM), Standard Type TM (STM) or Enhanced Type TM (ETM). Although similar in nature, the different TM types vary in their feature complexity. The common features to all of the Compact, Standard and Enhanced TMs will be described in this section and the detailed operation regarding each of the TM types will be described in separate sections. The main features and differences between the three types of TMs are summarised in the accompanying table.

Function	CTM	STM	ETM
Timer/Counter	√	√	√
I/P Capture	—	√	√
Compare Match Output	√	√	√
PWM Channels	1	1	2
Single Pulse Output	—	1	2
PWM Alignment	Edge	Edge	Edge & Centre
PWM Adjustment Period & Duty	Duty or Period	Duty or Period	Duty or Period

TM Function Summary

Each device in the series contains a specific number of either Compact Type, Standard Type and Enhanced Type TM units which are shown in the table together with their individual reference name, TM0~TM3.

Device	TM0	TM1	TM2	TM3
BC66F840	16-bit CTM	—	16-bit STM	16-bit ETM
BC66F850	16-bit CTM	16-bit CTM	16-bit STM	16-bit ETM
BC66F860	16-bit CTM	16-bit CTM	16-bit STM	16-bit ETM

TM Name/Type Reference

TM Operation

The three different types of TM offer a diverse range of functions, from simple timing operations to PWM signal generation. The key to understanding how the TM operates is to see it in terms of a free running counter whose value is then compared with the value of pre-programmed internal comparators. When the free running counter has the same value as the pre-programmed comparator, known as a compare match situation, a TM interrupt signal will be generated which can clear the counter and perhaps also change the condition of the TM output pin. The internal TM counter is driven by a user selectable clock source, which can be an internal clock or an external pin.

TM Clock Source

The clock source which drives the main counter in each TM can originate from various sources. The selection of the required clock source is implemented using the TnCK2~TnCK0 bits in the TM control registers. The clock source can be a ratio of either the system clock f_{SYS} or the internal high clock f_H, the f_{SUB} clock source or the external TCKn pin. Note that setting these bits to the value 101 will select a reserved clock input, in effect disconnecting the TM clock source. The TCKn pin clock source is used to allow an external signal to drive the TM as an external clock source or for event counting.

TM Interrupts

The Compact and Standard type TMs each have two internal interrupts, one for each of the internal comparator A or comparator P, which generate a TM interrupt when a compare match condition occurs. As the Enhanced type TM has three internal comparators and comparator A or comparator B or comparator P compare match functions, it consequently has three internal interrupts. When a TM interrupt is generated it can be used to clear the counter and also to change the state of the TM output pin.

TM External Pins

Each of the TMs, irrespective of what type, has one TM input pin, with the label TCKn. The TM input pin is essentially a clock source for the TM and is selected using the TnCK2~TnCK0 bits in the TMnC0 register. This external TM input pin allows an external clock source to drive the internal TM. This external TM input pin is shared with other functions but will be connected to the internal TM if selected using the TnCK2~TnCK0 bits. The TM input pin can be chosen to have either a rising or falling active edge.

The TMs each have one or more output pins with the label TPn. When the TM is in the Compare Match Output Mode, these pins can be controlled by the TM to switch to a high or low level or to toggle when a compare match situation occurs. The external TPn output pin is also the pin where the TM generates the PWM output waveform. As the TM output pins are pin-shared with other function, the TM output function must first be setup using registers. A single bit in one of the registers determines if its associated pin is to be used as an external TM output pin or if it is to have another function. The number of output pins for each TM type and device is different, the details are provided in the accompanying table.

Device	CTM	STM	ETM	Register
BC66F840	TP0	TP2	TP3A, TP3B	TMPC
BC66F850	TP0, TP1	TP2	TP3A, TP3B	TMPC
BC66F860	TP0, TP1	TP2	TP3A, TP3B	TMPC

TM Input/Output Pins

TM Input/Output Pin Control Registers

Selecting to have a TM input/output or whether to retain its other shared function is implemented using one register, with a single bit in each register corresponding to a TM input/output pin. Setting the bit high will setup the corresponding pin as a TM input/output, if reset to zero the pin will retain its original other function.

TMPC Register

Device	Bit							
	7	6	5	4	3	2	1	0
BC66F840	—	—	—	T3BCP	T3ACP	T2CP	—	T0CP
BC66F850	—	—	—	T3BCP	T3ACP	T2CP	T1CP	T0CP
BC66F860	—	—	—	T3BCP	T3ACP	T2CP	T1CP	T0CP

TM Input/Output Pin Control Register List

"–": Unimplemented, read as "0"

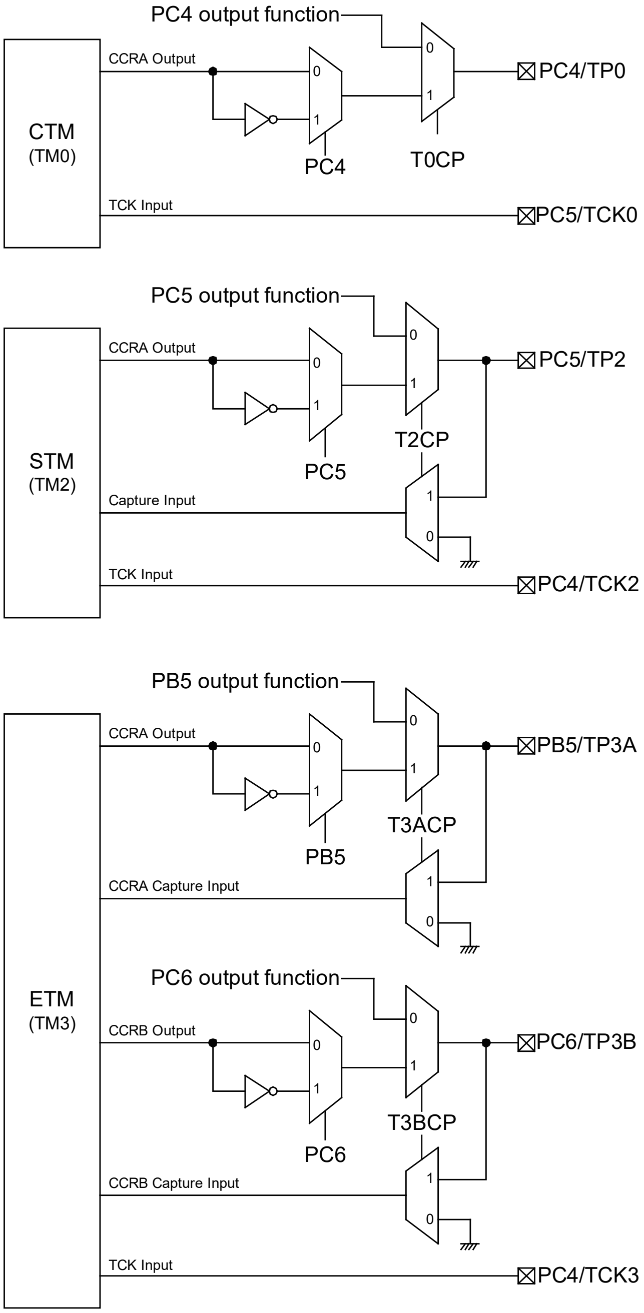
T3BCP: TP3B pin enable Control
0: Disable
1: Enable

T3ACP: TP3A pin enable Control
0: Disable
1: Enable

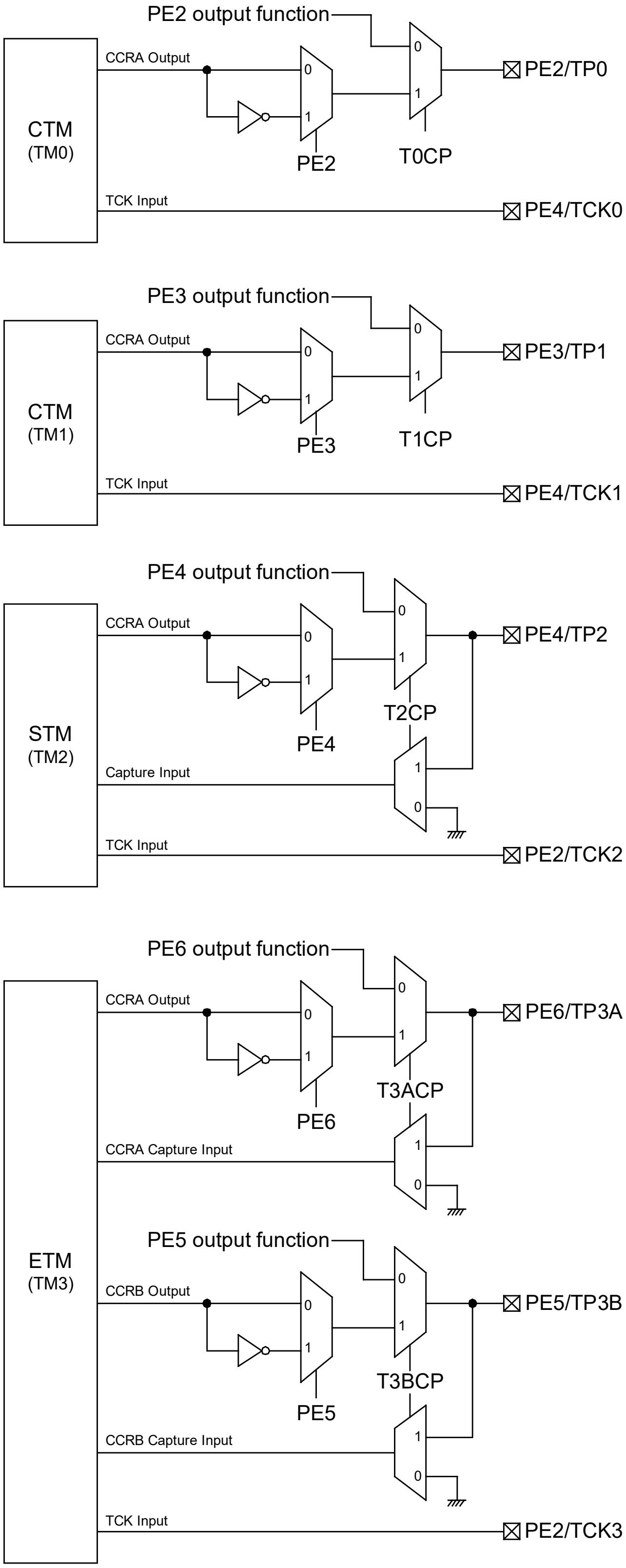
T2CP: TP2 pin enable Control
0: Disable
1: Enable

T1CP: TP1 pin enable Control
0: Disable
1: Enable

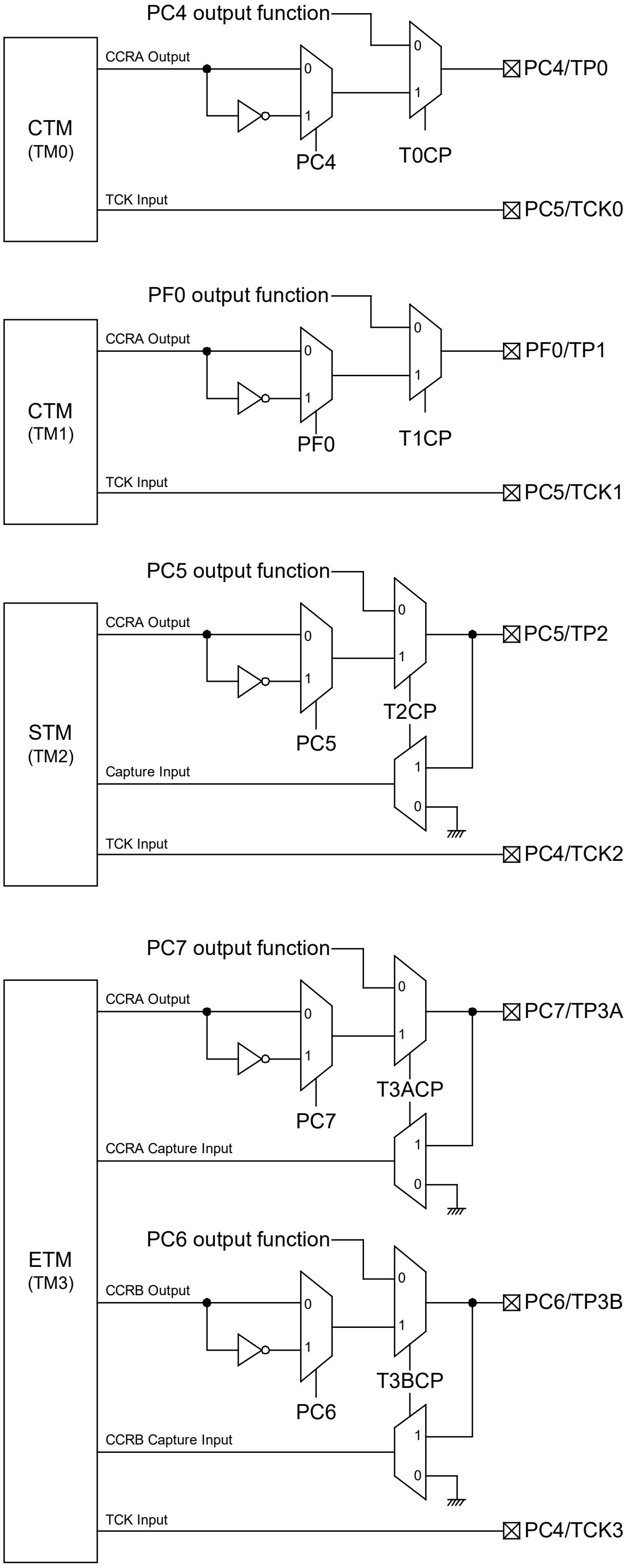
T0CP: TP0 pin enable Control
0: Disable
1: Enable



BC66F840 TM Function Pin Control Block Diagram



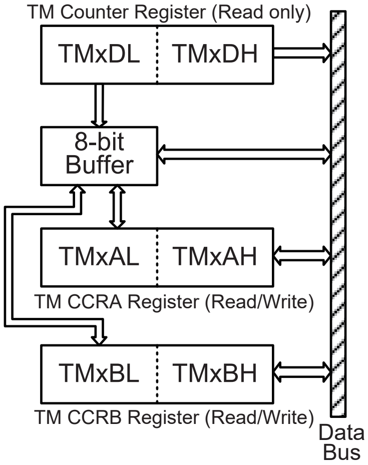
BC66F850 TM Function Pin Control Block Diagram



BC66F860 TM Function Pin Control Block Diagram

Programming Considerations

The TM Counter Registers and the Capture/Compare CCRA and CCRB registers, being 16-bit, all have a low and high byte structure. The high bytes can be directly accessed, but as the low bytes can only be accessed via an internal 8-bit buffer, reading or writing to these register pairs must be carried out in a specific way. The important point to note is that data transfer to and from the 8-bit buffer and its related low byte only takes place when a write or read operation to its corresponding high byte is executed. As the CCRA and CCRB registers are implemented in the way shown in the following diagram and accessing these register pairs is carried out in a specific way described above, it is recommended to use the "MOV" instruction to access the CCRA and CCRB low byte registers, named TMxAL and TMxBL, using the following access procedures. Accessing the CCRA or CCRB low byte registers without following these access procedures will result in unpredictable values.


TM Counter Register (Read only)
TMxDL
TMxDH
8-bit Buffer
TMxAL
TMxAH
TM CCRA Register (Read/Write)
TMxBL
TMxBH
TM CCRB Register (Read/Write)
Data Bus

The following steps show the read and write procedures:

- Writing Data to CCRB or CCRA
    - Write data to Low Byte TMxAL or TMxBL – note that here data is only written to the 8-bit buffer.
    - Write data to High Byte TMxAH or TMxBH – here data is written directly to the high byte registers and simultaneously data is latched from the 8-bit buffer to the Low Byte registers.
- Reading Data from the Counter Registers and CCRB or CCRA
    - Read data from the High Byte TMxDH, TMxAH or TMxBH – here data is read directly from the High Byte registers and simultaneously data is latched from the Low Byte register into the 8-bit buffer.
    - Read data from the Low Byte TMxDL, TMxAL or TMxBL – this step reads data from the 8-bit buffer.

Compact Type TM – CTM

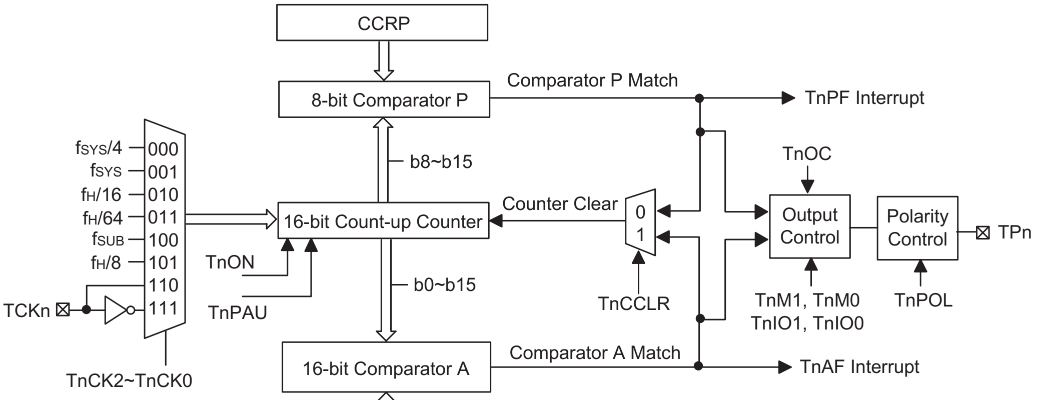
Although the simplest form of the three TM types, the Compact TM type still contains three operating modes, which are Compare Match Output, Timer/Event Counter and PWM Output modes. The Compact TM can also be controlled with an external input pin and can drive one external output pin.

CTM	Name	TM No.	TM Input Pin	TM Output Pin
BC66F840	16-bit CTM	0	TCK0	TP0
BC66F850/ BC66F860	16-bit CTM	0, 1	TCK0, TCK1	TP0, TP1

Compact TM Operation

At its core is a 16-bit count-up counter which is driven by a user selectable internal or external clock source. There are also two internal comparators with the names, Comparator A and Comparator P. These comparators will compare the value in the counter with CCRP and CCRAregisters. The CCRP is 8-bit wide whose value is compared with the highest eight bits in the counter while the CCRA is 16-bit wide and therefore compares with all counter bits.

The only way of changing the value of the 16-bit counter using the application program, is to clear the counter by changing the TnON bit from low to high. The counter will also be cleared automatically by a counter overflow or a compare match with one of its associated comparators. When these conditions occur, a TM interrupt signal will also usually be generated. The Compact Type TM can operate in a number of different operational modes, can be driven by different clock sources including an input pin and can also control an output pin. All operating setup conditions are selected using relevant internal registers.



Compact Type TM Block Diagram (n=0 or 1)

Compact Type TM Register Description

Overall operation of the Compact TM is controlled using a series of registers. Aread only register pair exists to store the internal counter 16-bit value, while a read/write register pair exists to store the internal 16-bit CCRA value. There is also a read/write register used to store the internal 8-bit CCRP value. The remaining two registers are control registers which setup the different operating and control modes.

Register Name	Bit							
	7	6	5	4	3	2	1	0
TMnC0	TnPAU	TnCK2	TnCK1	TnCK0	TnON	—	—	—
TMnC1	TnM1	TnM0	TnIO1	TnIO0	TnOC	TnPOL	TnDPX	TnCCLR
TMnDL	D7	D6	D5	D4	D3	D2	D1	D0
TMnDH	D15	D14	D13	D12	D11	D10	D9	D8
TMnAL	D7	D6	D5	D4	D3	D2	D1	D0
TMnAH	D15	D14	D13	D12	D11	D10	D9	D8
TMnRP	TnRP7	TnRP6	TnRP5	TnRP4	TnRP3	TnRP2	TnRP1	TnRP0

(n=0 for BC66F840 while n=0~1 for BC66F850/BC66F860)

16-bit Compact TM Register List

TMnDL Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TMnDL:** TMn Counter Low Byte Register bit 7~bit 0
TMn 16-bit Counter bit 7~bit 0

TMnDH Register

Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TMnDH:** TMn Counter High Byte Register bit 7~bit 0
TMn 16-bit Counter bit 15~bit 8

TMnAL Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TMnAL:** TMn CCRA Low Byte Register bit 7~bit 0
TMn 16-bit CCRA bit 7~bit 0

TMnAH Register

Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TMnAH:** TMn CCRA High Byte Register bit 7~bit 0
TMn 16-bit CCRA bit 15~bit 8

TMnC0 Register

Bit	7	6	5	4	3	2	1	0
Name	TnPAU	TnCK2	TnCK1	TnCK0	TnON	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

Bit 7 **TnPAU:** TMn Counter Pause Control
0: Run
1: Pause

The counter can be paused by setting this bit high. Clearing the bit to zero restores normal counter operation. When in a Pause condition the TM will remain powered up and continue to consume power. The counter will retain its residual value when this bit changes from low to high and resume counting from this value when the bit changes to a low value again.

Bit 6~4 **TnCK2~TnCK0:** Select TMn Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: $f_H/8$
110: TCKn rising edge clock
111: TCKn falling edge clock

These three bits are used to select the clock source for the TM. Selecting the Reserved clock input will effectively disable the internal counter. The external pin clock source can be chosen to be active on the rising or falling edge. The clock source f_{SYS} is the system clock, while f_H and f_{SUB} are other internal clocks, the details of which can be found in the oscillator section.

Bit 3 **TnON:** TMn Counter On/Off Control
0: Off
1: On

This bit controls the overall on/off function of the TM. Setting the bit high enables the counter to run and clearing the bit disables the TM. Clearing this bit to zero will stop the counter from counting and turn off the TM which will reduce its power consumption. When the bit changes state from low to high the internal counter value will be reset to zero, however when the bit changes from high to low, the internal counter will retain its residual value.

If the TM is in the Compare Match Output Mode then the TM output pin will be reset to its initial condition, as specified by the TnOC bit, when the TnON bit changes from low to high.

Bit 2~0 Unimplemented, read as “0”

TMnC1 Register

Bit	7	6	5	4	3	2	1	0
Name	TnM1	TnM0	TnIO1	TnIO0	TnOC	TnPOL	TnDPX	TnCCLR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **TnM1~TnM0:** Select TMn Operation Mode
00: Compare Match Output Mode
01: Undefined
10: PWM Mode
11: Timer/Counter Mode

These bits setup the required operating mode for the TM. To ensure reliable operation the TM should be switched off before any changes are made to the TnM1 and TnM0 bits. In the Timer/Counter Mode, the TM output pin control must be disabled.

Bit 5~4 **TnIO1~TnIO0:** Select TPn output function
Compare Match Output Mode
00: No change
01: Output low
10: Output high
11: Toggle output
PWM Mode
00: PWM Output inactive state
01: PWM Output active state
10: PWM output
11: Undefined
Timer/Counter Mode
Unused

These two bits are used to determine how the TM output pin changes state when a certain condition is reached. The function that these bits select depends upon in which mode the TM is running.

In the Compare Match Output Mode, the TnIO1 and TnIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the TnOC bit in the TMnC1 register. Note that the output level requested by the TnIO1 and TnIO0 bits must be different from the initial value setup using the TnOC bit otherwise no change will occur on the TM output pin when a compare match occurs. After the TM output pin changes state, it can be reset to its initial level by changing the level of the TnON bit from low to high.

In the PWM Mode, the TnIO1 and TnIO0 bits determine how the TM output pin changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to change the values of the TnIO1 and TnIO0 bits only after the TMn has been switched off. Unpredictable PWM outputs will occur if the TnIO1 and TnIO0 bits are changed when the TM is running.

Bit 3 **TnOC:** TPn output control bit
Compare Match Output Mode
0: Initial low
1: Initial high
PWM Mode
0: Active low
1: Active high

This is the output control bit for the TM output pin. Its operation depends upon whether TM is being used in the Compare Match Output Mode or in the PWM Mode. It has no effect if the TM is in the Timer/Counter Mode. In the Compare Match Output Mode it determines the logic level of the TM output pin before a compare match occurs. In the PWM Mode it determines if the PWM signal is active high or active low.

Bit 2 **TnPOL:** TPn output Polarity control
0: Non-invert
1: Invert

This bit controls the polarity of the TPn output pin. When the bit is set high the TM output pin will be inverted and not inverted when the bit is zero. It has no effect if the TM is in the Timer/Counter Mode.

Bit 1 **TnDPX:** TMn PWM period/duty Control
0: CCRP – period; CCRA – duty
1: CCRP – duty; CCRA – period

This bit, determines which of the CCRA and CCRP registers are used for period and duty control of the PWM waveform.

Bit 0 **TnCCLR:** Select TMn Counter clear condition
0: TMn Comparator P match
1: TMn Comparator A match

This bit is used to select the method which clears the counter. Remember that the Compact TM contains two comparators, Comparator A and Comparator P, either of which can be selected to clear the internal counter. With the TnCCLR bit set high, the counter will be cleared when a compare match occurs from the Comparator A. When the bit is low, the counter will be cleared when a compare match occurs from the Comparator P or with a counter overflow. A counter overflow clearing method can only be implemented if the CCRP bits are all cleared to zero. The TnCCLR bit is not used in the PWM Mode.

TMnRP Register

Bit	7	6	5	4	3	2	1	0
Name	TnRP7	TnRP6	TnRP5	TnRP4	TnRP3	TnRP2	TnRP1	TnRP0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TnRP7~TnRP0:** TMn CCRP register bit 7~bit 0, compare with the TMn counter bit 15~bit 8
Comparator P Match Period
0: 65536 TMn clocks
1~255: (1~255)×256 TMn clocks

These eight bits are used to setup the value on the internal CCRP 8-bit register, which are then compared with the internal counter's highest eight bits. The result of this comparison can be selected to clear the internal counter if the TnCCLR bit is set to zero. Setting the TnCCLR bit to zero ensures that a compare match with the CCRP values will reset the internal counter. As the CCRP bits are only compared with the highest eight counter bits, the compare values exist in 256 clock cycle multiples. Clearing all eight bits to zero is in effect allowing the counter to overflow at its maximum value.

Compact Type TM Operation Modes

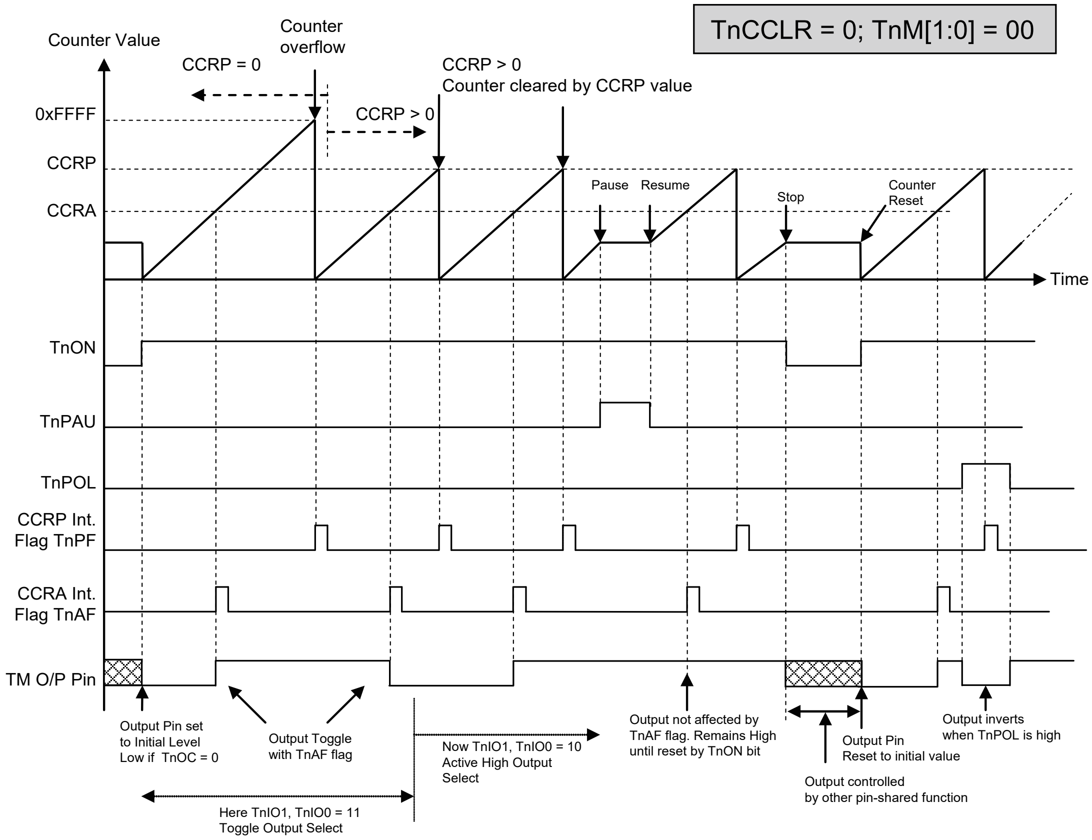
The Compact Type TM can operate in one of three operating modes, Compare Match Output Mode, PWM Mode or Timer/Counter Mode. The operating mode is selected using the TnM1 and TnM0 bits in the TMnC1 register.

Compare Match Output Mode

To select this mode, bits TnM1 and TnM0 in the TMnC1 register, should be set to 00B respectively. In this mode once the counter is enabled and running it can be cleared by three methods. These are a counter overflow, a compare match from Comparator A and a compare match from Comparator P. When the TnCCLR bit is low, there are two ways in which the counter can be cleared. One is when a compare match occurs from Comparator P, the other is when the CCRP bits are all zero which allows the counter to overflow. Here both TnAF and TnPF interrupt request flags for the Comparator A and Comparator P respectively, will both be generated.

If the TnCCLR bit in the TMnC1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the TnAF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when TnCCLR is high no TnPF interrupt request flag will be generated. If the CCRA bits are all zero, the counter will overflow when its reaches its maximum 16-bit, FFFF Hex, value, however here the TnAF interrupt request flag will not be generated.

As the name of the mode suggests, after a comparison is made, the TM output pin will change state. The TM output pin condition however only changes state when a TnAF interrupt request flag is generated after a compare match occurs from Comparator A. The TnPF interrupt request flag, generated from a compare match occurs from Comparator P, will have no effect on the TM output pin. The way in which the TM output pin changes state are determined by the condition of the TnIO1 and TnIO0 bits in the TMnC1 register. The TM output pin can be selected using the TnIO1 and TnIO0 bits to go high, to go low or to toggle from its present condition when a compare match occurs from Comparator A. The initial condition of the TM output pin, which is setup after the TnON bit changes from low to high, is setup using the TnOC bit. Note that if the TnIO1 and TnIO0 bits are zero then no pin change will take place.


TnCCLR = 0; TnM[1:0] = 00
Counter Value
Counter overflow
CCRP = 0
CCRP > 0
Counter cleared by CCRP value
0xFFFF
CCRP > 0
CCRP
CCRA
Pause
Resume
Stop
Counter Reset
Time
TnON
TnPAU
TnPOL
CCRP Int. Flag TnPF
CCRA Int. Flag TnAF
TM O/P Pin
Output Pin set to Initial Level Low if TnOC = 0
Output Toggle with TnAF flag
Now TnIO1, TnIO0 = 10 Active High Output Select
Output not affected by TnAF flag. Remains High until reset by TnON bit
Output Pin Reset to initial value
Output inverts when TnPOL is high
Output controlled by other pin-shared function
Here TnIO1, TnIO0 = 11 Toggle Output Select

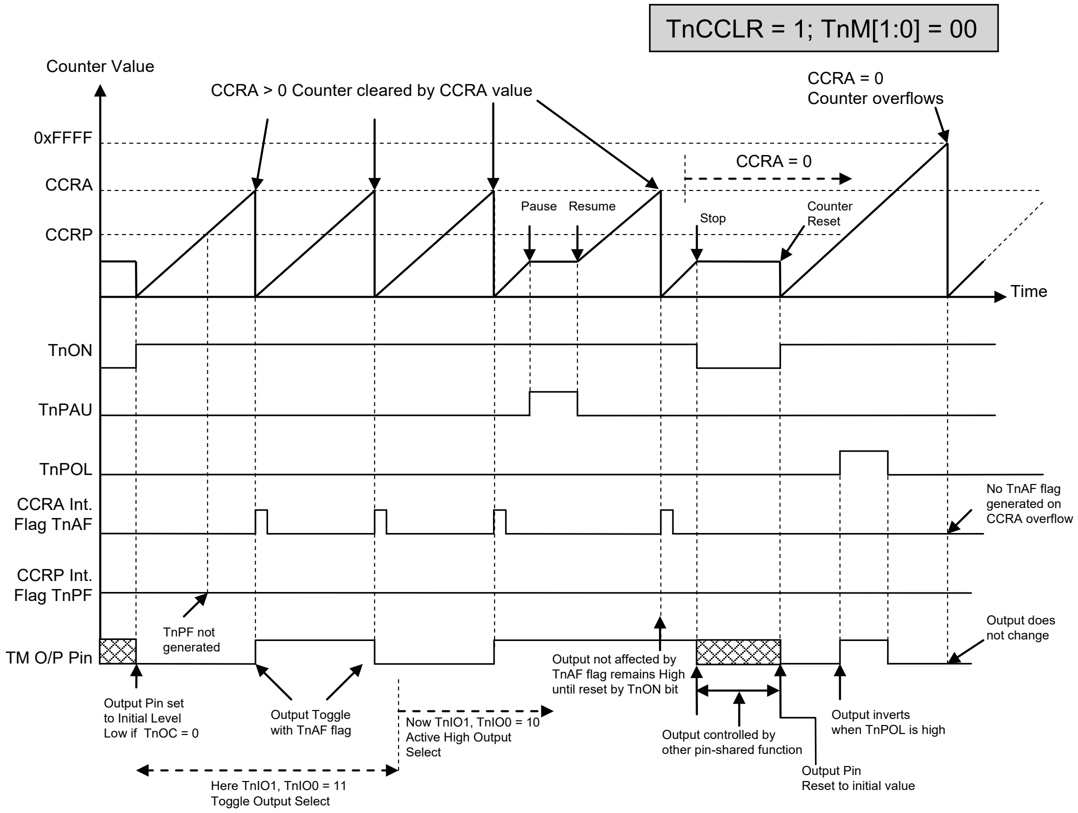
Compare Match Output Mode – TnCCLR=0

Note: 1. With TnCCLR=0, a Comparator P match will clear the counter

2. The TM output pin is controlled only by the TnAF flag

3. The output pin is reset to its initial state by a TnON bit rising edge

4. n=0, 1



Compare Match Output Mode – TnCCLR=1

Note: 1. With TnCCLR=1, a Comparator A match will clear the counter

2. The TM output pin is controlled only by the TnAF flag

3. The output pin is reset to its initial state by a TnON bit rising edge

4. The TnPF flag is not generated when TnCCLR=1

5. n=0, 1

Timer/Counter Mode

To select this mode, bits TnM1 and TnM0 in the TMnC1 register should be set to 11 respectively. The Timer/Counter Mode operates in an identical way to the Compare Match Output Mode generating the same interrupt flags. The exception is that in the Timer/Counter Mode the TM output pin is not used. Therefore the above description and Timing Diagrams for the Compare Match Output Mode can be used to understand its function. As the TM output pin is not used in this mode, the pin can be used as a normal I/O pin or other pin-shared function.

PWM Output Mode

To select this mode, bits TnM1 and TnM0 in the TMnC1 register should be set to 10 respectively. The PWM function within the TM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the TM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM mode, the TnCCLR bit has no effect on the PWM operation. Both of the CCRA and CCRP registers are used to generate the PWM waveform, one register is used to clear the internal counter and thus control the PWM waveform frequency, while the other one is used to control the duty cycle. Which register is used to control either frequency or duty cycle is determined using the TnDPX bit in the TMnC1 register. The PWM waveform frequency and duty cycle can therefore be controlled by the values in the CCRA and CCRP registers.

An interrupt flag, one for each of the CCRA and CCRP, will be generated when a compare match occurs from either Comparator A or Comparator P. The TnOC bit in the TMnC1 register is used to select the required polarity of the PWM waveform while the two TnIO1 and TnIO0 bits are used to enable the PWM output or to force the TM output pin to a fixed high or low level. The TnPOL bit is used to reverse the polarity of the PWM output waveform.

16-bit CTM, PWM Mode, Edge-aligned Mode, TnDPX=0

CCRP	1~255	0
Period	CCRP×256	65536
Duty	CCRA	

If f_{SYS}=16MHz, TM clock source is f_{SYS}/4, CCRP=2 and CCRA=128

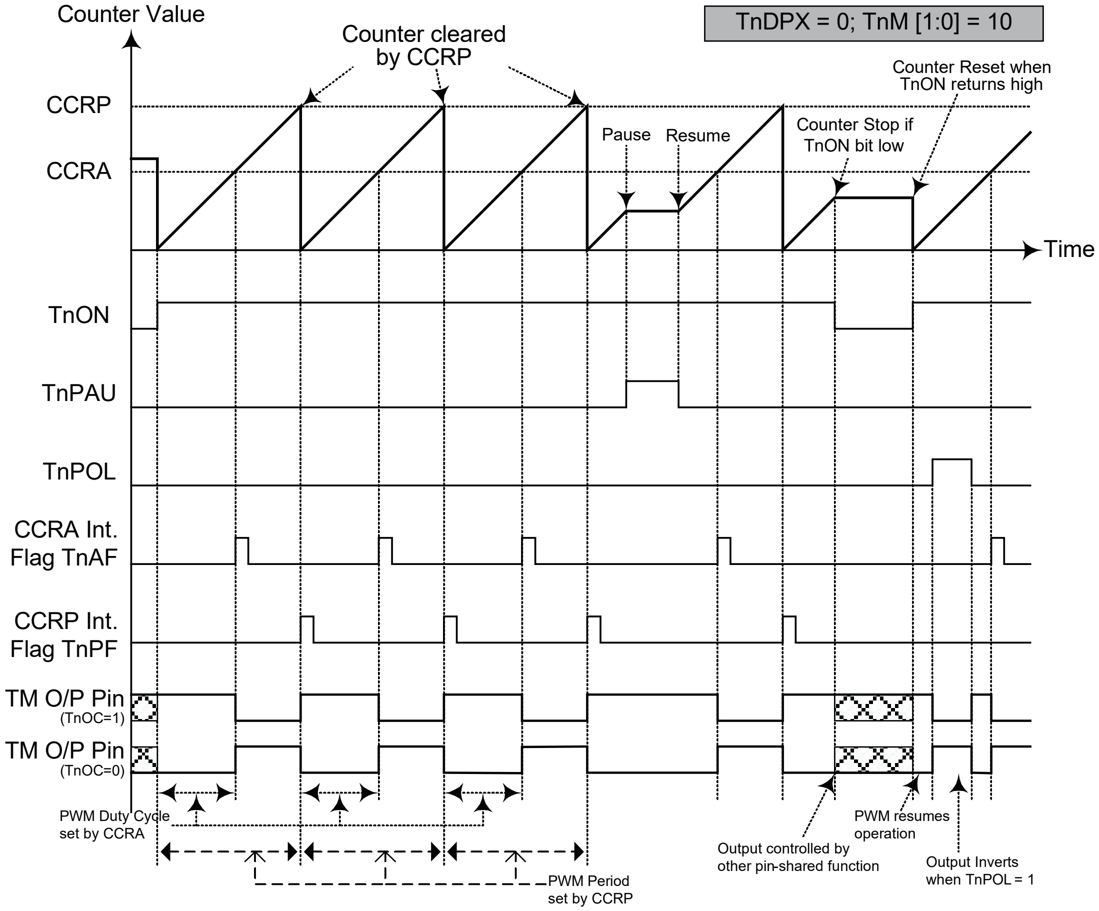
The CTM PWM output frequency=(f_{SYS}/4)/(2×256)=f_{SYS}/2048=7.8125kHz, duty=128/ (2×256)=25%.

If the Duty value defined by the CCRA register is equal to or greater than the Period value, then the PWM output duty is 100%.

16-bit CTM, PWM Mode, Edge-aligned Mode, TnDPX=1

CCRP	1~255	0
Period	CCRA	
Duty	CCRP×256	65536

The PWM output period is determined by the CCRA register value together with the TM clock while the PWM duty cycle is defined by the (CCRP×256) except when the CCRP value is equal to 0.



PWM Mode – TnDPX=0

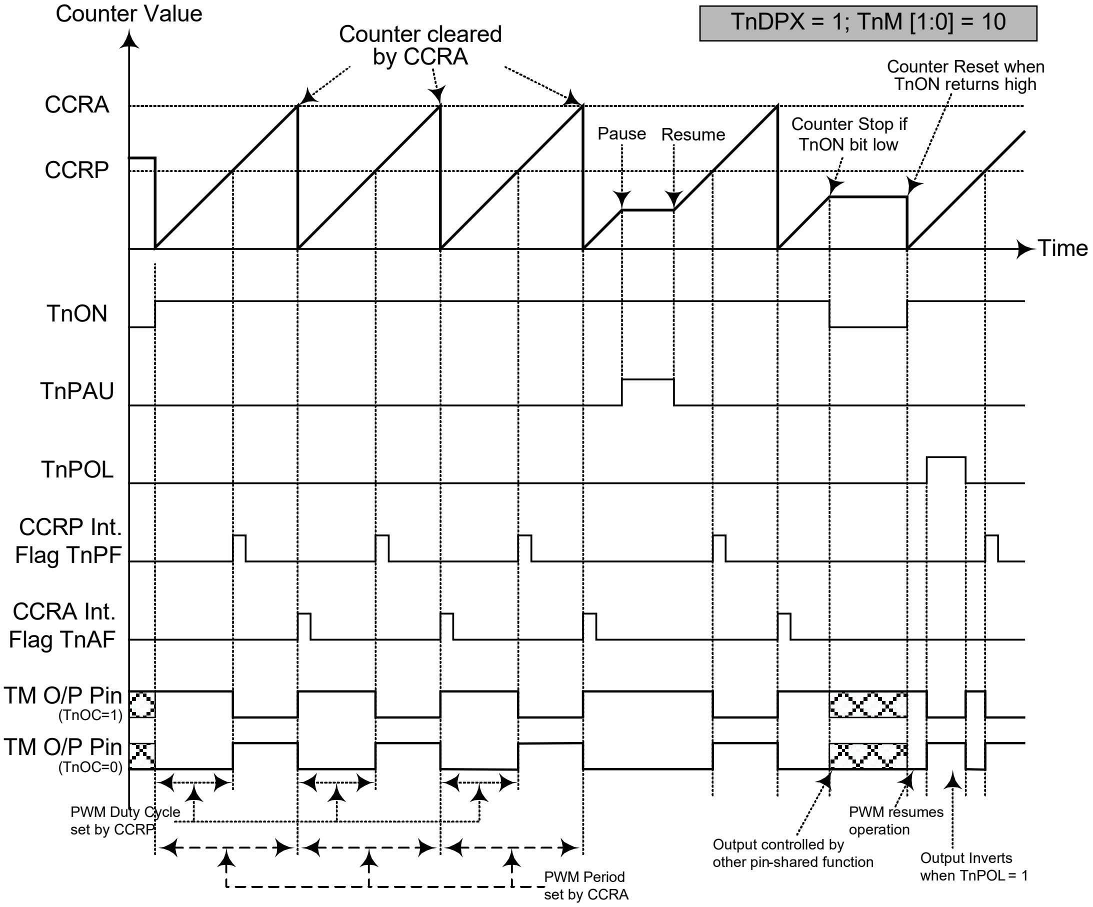
Note: 1. Here TnDPX=0 – Counter cleared by CCRP

2. A counter clear sets the PWM Period

3. The internal PWM function continues even when TnIO [1:0]=00 or 01

4. The TnCCLR bit has no influence on PWM operation

5. n=0, 1



PWM Mode – TnDPX=1

Note: 1. Here TnDPX=1 – Counter cleared by CCRA

2. A counter clear sets the PWM Period

3. The internal PWM function continues even when TnIO [1:0]=00 or 01

4. The TnCCLR bit has no influence on PWM operation

5. n=0, 1

Standard Type TM – STM

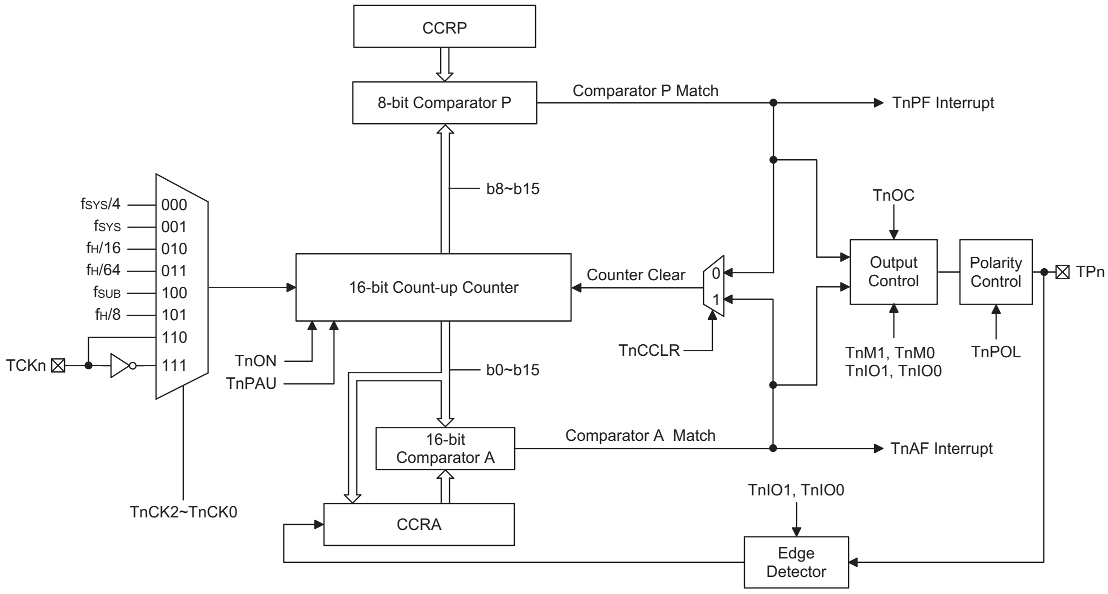
The Standard Type TM contains five operating modes, which are Compare Match Output, Timer/Event Counter, Capture Input, Single Pulse Output and PWM Output modes. The Standard TM can also be controlled with an external input pin and can drive one external output pin.

STM	Name	TM No.	TM Input Pin	TM Output Pin
BC66F840 BC66F850 BC66F860	16-bit STM	2	TCK2	TP2

Standard TM Operation

At its core is a 16-bit count-up counter which is driven by a user selectable internal or external clock source. There are also two internal comparators with the names, Comparator A and Comparator P. These comparators will compare the value in the counter with CCRP and CCRA registers. The CCRP comparator is 8-bit wide whose value is compared with the highest eight bits in the counter while the CCRA is 16-bit wide and therefore compares with all counter bits.

The only way of changing the value of the 16-bit counter using the application program, is to clear the counter by changing the TnON bit from low to high. The counter will also be cleared automatically by a counter overflow or a compare match with one of its associated comparators. When these conditions occur, a TM interrupt signal will also usually be generated. The Standard Type TM can operate in a number of different operational modes, can be driven by different clock sources including an input pin and can also control an output pin. All operating setup conditions are selected using relevant internal registers.



Standard Type TM Block Diagram (n=2)

Standard Type TM Register Description

Overall operation of the Standard TM is controlled using a series of registers. A read only register pair exists to store the internal counter 16-bit value, while a read/write register pair exists to store the internal 16-bit CCRA value. There is also a read/write register used to store the internal 8-bit CCRP value. The remaining two registers are control registers which setup the different operating and control modes.

Register Name	Bit							
	7	6	5	4	3	2	1	0
TMnC0	TnPAU	TnCK2	TnCK1	TnCK0	TnON	—	—	—
TMnC1	TnM1	TnM0	TnIO1	TnIO0	TnOC	TnPOL	TnDPX	TnCCLR
TMnDL	D7	D6	D5	D4	D3	D2	D1	D0
TMnDH	D15	D14	D13	D12	D11	D10	D9	D8
TMnAL	D7	D6	D5	D4	D3	D2	D1	D0
TMnAH	D15	D14	D13	D12	D11	D10	D9	D8
TMnRP	TnRP7	TnRP6	TnRP5	TnRP4	TnRP3	TnRP2	TnRP1	TnRP0

16-bit Standard TM Register List (n=2)

TMnDL Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TMnDL:** TMn Counter Low Byte Register bit 7~bit 0
TMn 16-bit Counter bit 7~bit 0

TMnDH Register

Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TMnDH:** TMn Counter High Byte Register bit 7~bit 0
TMn 16-bit Counter bit 15~bit 8

TMnAL Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TMnAL:** TMn CCRA Low Byte Register bit 7~bit 0
TMn 16-bit CCRA bit 7~bit 0

TMnAH Register

Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TMnAH:** TMn CCRA High Byte Register bit 7~bit 0
TMn 16-bit CCRA bit 15~bit 8

TMnC0 Register

Bit	7	6	5	4	3	2	1	0
Name	TnPAU	TnCK2	TnCK1	TnCK0	TnON	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

Bit 7 **TnPAU:** TMn Counter Pause Control
0: Run
1: Pause

The counter can be paused by setting this bit high. Clearing the bit to zero restores normal counter operation. When in a Pause condition the TM will remain powered up and continue to consume power. The counter will retain its residual value when this bit changes from low to high and resume counting from this value when the bit changes to a low value again.

Bit 6~4 **TnCK2~TnCK0:** Select TMn Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: $f_H/8$
110: TCKn rising edge clock
111: TCKn falling edge clock

These three bits are used to select the clock source for the TM. Selecting the Reserved clock input will effectively disable the internal counter. The external pin clock source can be chosen to be active on the rising or falling edge. The clock source f_{SYS} is the system clock, while f_H and f_{SUB} are other internal clocks, the details of which can be found in the oscillator section.

Bit 3 **TnON:** TMn Counter On/Off Control
0: Off
1: On

This bit controls the overall on/off function of the TM. Setting the bit high enables the counter to run and clearing the bit disables the TM. Clearing this bit to zero will stop the counter from counting and turn off the TM which will reduce its power consumption. When the bit changes state from low to high the internal counter value will be reset to zero, however when the bit changes from high to low, the internal counter will retain its residual value.

If the TM is in the Compare Match Output Mode then the TM output pin will be reset to its initial condition, as specified by the TnOC bit, when the TnON bit changes from low to high.

Bit 2~0 Unimplemented, read as “0”

TMnC1 Register

Bit	7	6	5	4	3	2	1	0
Name	TnM1	TnM0	TnIO1	TnIO0	TnOC	TnPOL	TnDPX	TnCCLR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **TnM1~TnM0:** Select TMn Operation Mode
00: Compare Match Output Mode
01: Capture Input Mode
10: PWM Mode or Single Pulse Output Mode
11: Timer/Counter Mode

These bits setup the required operating mode for the TM. To ensure reliable operation the TM should be switched off before any changes are made to the TnM1 and TnM0 bits. In the Timer/Counter Mode, the TM output pin control must be disabled.

Bit 5~4 **TnIO1~TnIO0:** Select TPn output function
Compare Match Output Mode
00: No change
01: Output low
10: Output high
11: Toggle output
PWM Mode/Single Pulse Output Mode
00: PWM Output inactive state
01: PWM Output active state
10: PWM output
11: Single pulse output
Capture input Mode
00: Input capture at rising edge of TPn
01: Input capture at falling edge of TPn
01: Input capture at falling/rising edge of TPn
11: Input capture disabled
Timer/Counter Mode
Unused

These two bits are used to determine how the TM output pin changes state when a certain condition is reached. The function that these bits select depends upon in which mode the TM is running.

In the Compare Match Output Mode, the TnIO1 and TnIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the TnOC bit in the TMnC1 register. Note that the output level requested by the TnIO1 and TnIO0 bits must be different from the initial value setup using the TnOC bit otherwise no change will occur on the TM output pin when a compare match occurs. After the TM output pin changes state, it can be reset to its initial level by changing the level of the TnON bit from low to high.

In the PWM Mode, the TnIO1 and TnIO0 bits determine how the TM output pin changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to change the values of the TnIO1 and TnIO0 bits only after the TM has been switched off. Unpredictable PWM outputs will occur if the TnIO1 and TnIO0 bits are changed when the TM is running.

Bit 3 **TnOC:** TPn output control bit
Compare Match Output Mode
0: Initial low
1: Initial high
PWM Mode/Single Pulse Output Mode
0: Active low
1: Active high

This is the output control bit for the TM output pin. Its operation depends upon whether TM is being used in the Compare Match Output Mode or in the PWM Mode/Single Pulse Output Mode. It has no effect if the TM is in the Timer/Counter Mode. In the Compare Match Output Mode it determines the logic level of the TM output pin before a compare match occurs. In the PWM Mode it determines if the PWM signal is active high or active low.

Bit 2 **TnPOL:** TPn output Polarity control
0: Non-invert
1: Invert

This bit controls the polarity of the TPn output pin. When the bit is set high the TM output pin will be inverted and not inverted when the bit is zero. It has no effect if the TM is in the Timer/Counter Mode.

Bit 1 **TnDPX:** TMn PWM period/duty Control
0: CCRP – period; CCRA – duty
1: CCRP – duty; CCRA – period

This bit, determines which of the CCRA and CCRP registers are used for period and duty control of the PWM waveform.

Bit 0 **TnCCLR:** Select TMn Counter clear condition
0: TMn Comparator P match
1: TMn Comparator A match

This bit is used to select the method which clears the counter. Remember that the Standard TM contains two comparators, Comparator A and Comparator P, either of which can be selected to clear the internal counter. With the TnCCLR bit set high, the counter will be cleared when a compare match occurs from the Comparator A. When the bit is low, the counter will be cleared when a compare match occurs from the Comparator P or with a counter overflow. A counter overflow clearing method can only be implemented if the CCRP bits are all cleared to zero. The TnCCLR bit is not used in the PWM Mode, Single Pulse or Input Capture Mode.

TMnRP Register

Bit	7	6	5	4	3	2	1	0
Name	TnRP7	TnRP6	TnRP5	TnRP4	TnRP3	TnRP2	TnRP1	TnRP0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TnRP7~TnRP0:** TMn CCRP register bit 7~bit 0, compare with the TMn counter bit 15~bit 8
Comparator P Match Period
0: 65536 TMn clocks
1~255: (1~255)×256 TMn clocks

These eight bits are used to setup the value on the internal CCRP 8-bit register, which are then compared with the internal counter's highest eight bits. The result of this comparison can be selected to clear the internal counter if the TnCCLR bit is set to zero. Setting the TnCCLR bit to zero ensures that a compare match with the CCRP values will reset the internal counter. As the CCRP bits are only compared with the highest eight counter bits, the compare values exist in 256 clock cycle multiples. Clearing all eight bits to zero is in effect allowing the counter to overflow at its maximum value.

Standard Type TM Operation Modes

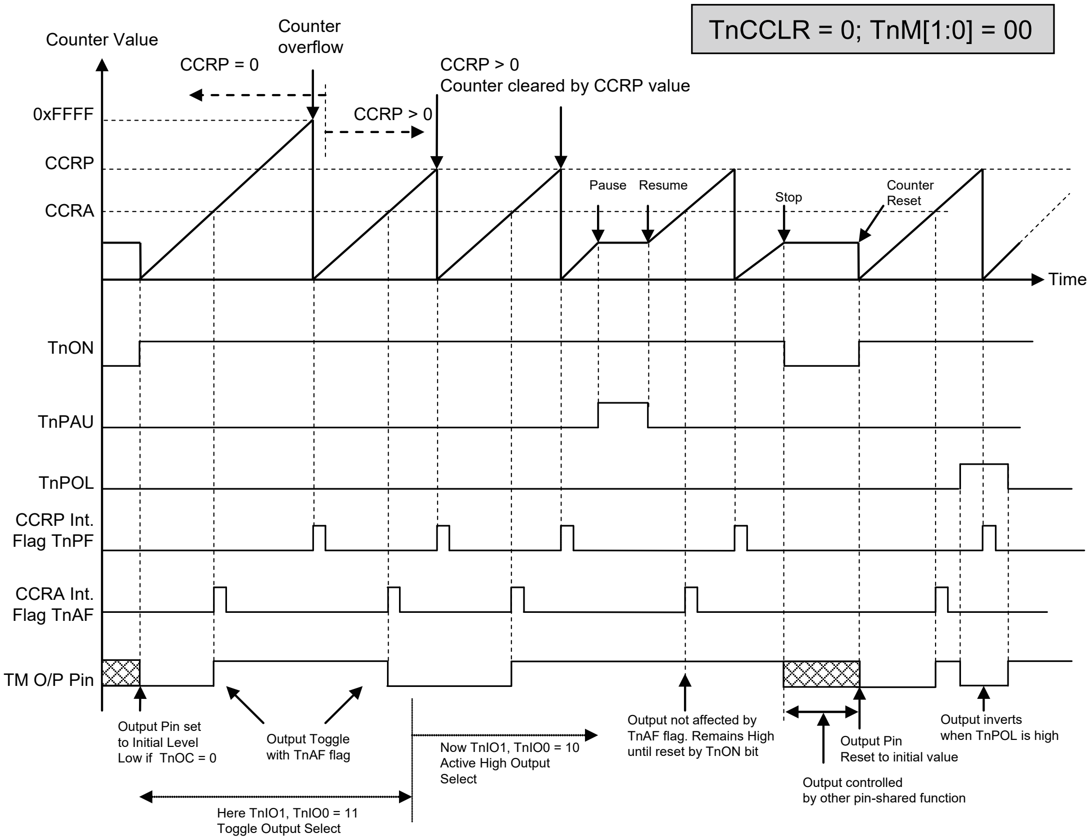
The Standard Type TM can operate in one of five operating modes, Compare Match Output Mode, PWM Mode, Single Pulse Output Mode, Capture Input Mode or Timer/Counter Mode. The operating mode is selected using the TnM1 and TnM0 bits in the TMnC1 register.

Compare Match Output Mode

To select this mode, bits TnM1 and TnM0 in the TMnC1 register, should be set to 00 respectively. In this mode once the counter is enabled and running it can be cleared by three methods. These are a counter overflow, a compare match from Comparator A and a compare match from Comparator P. When the TnCCLR bit is low, there are two ways in which the counter can be cleared. One is when a compare match from Comparator P, the other is when the CCRP bits are all zero which allows the counter to overflow. Here both TnAF and TnPF interrupt request flags for Comparator Aand Comparator P respectively, will both be generated.

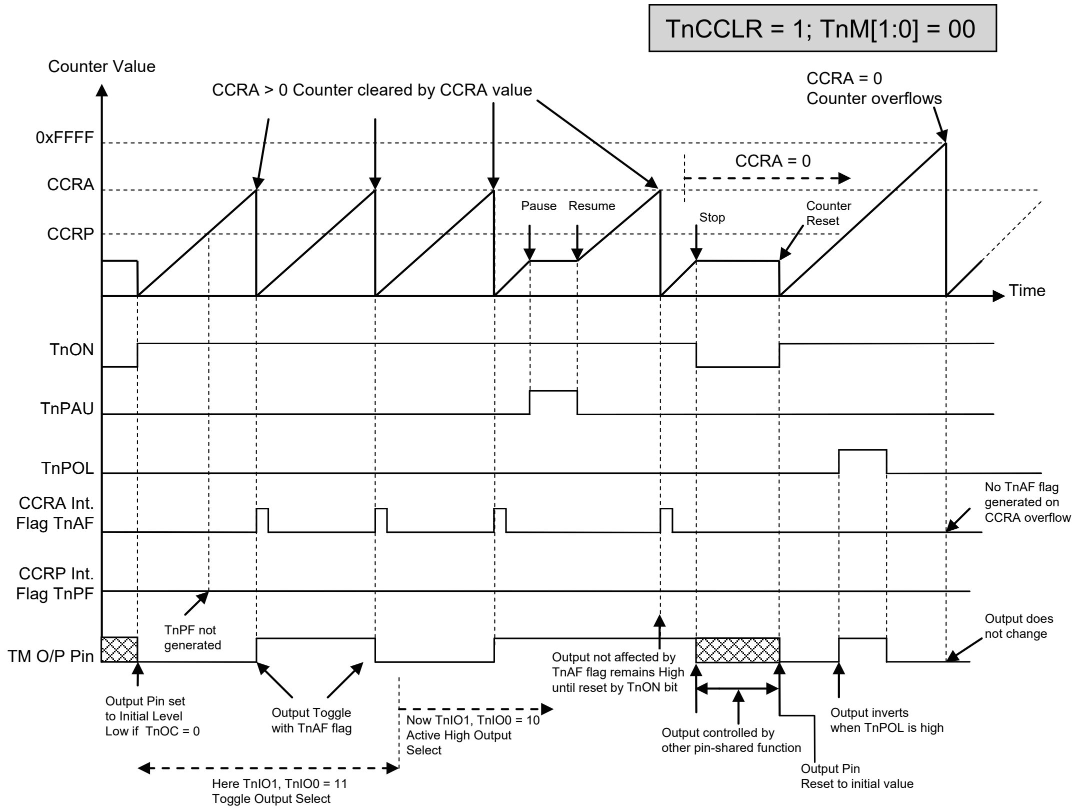
If the TnCCLR bit in the TMnC1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the TnAF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when TnCCLR is high no TnPF interrupt request flag will be generated. In the Compare Match Output Mode, the CCRA can not be set to 0.

As the name of the mode suggests, after a comparison is made, the TM output pin, will change state. The TM output pin condition however only changes state when a TnAF interrupt request flag is generated after a compare match occurs from Comparator A. The TnPF interrupt request flag, generated from a compare match occurs from Comparator P, will have no effect on the TM output pin. The way in which the TM output pin changes state are determined by the condition of the TnIO1 and TnIO0 bits in the TMnC1 register. The TM output pin can be selected using the TnIO1 and TnIO0 bits to go high, to go low or to toggle from its present condition when a compare match occurs from Comparator A. The initial condition of the TM output pin, which is setup after the TnON bit changes from low to high, is setup using the TnOC bit. Note that if the TnIO1 and TnIO0 bits are zero then no pin change will take place.


TnCCLR = 0; TnM[1:0] = 00
Counter Value
Counter overflow
CCRP = 0
CCRP > 0
CCRP > 0
Counter cleared by CCRP value
0xFFFF
CCRP
CCRA
Pause
Resume
Stop
Counter Reset
Time
TnON
TnPAU
TnPOL
CCRP Int. Flag TnPF
CCRA Int. Flag TnAF
TM O/P Pin
Output Pin set to Initial Level Low if TnOC = 0
Output Toggle with TnAF flag
Now TnIO1, TnIO0 = 10 Active High Output Select
Output not affected by TnAF flag. Remains High until reset by TnON bit
Output Pin Reset to initial value
Output inverts when TnPOL is high
Output controlled by other pin-shared function
Here TnIO1, TnIO0 = 11 Toggle Output Select

Compare Match Output Mode – TnCCLR=0

Note: 1. With TnCCLR=0, A Comparator P match will clear the counter
2. The TM output pin is controlled only by the TnAF flag
3. The output pin is reset to its initial state by a TnON bit rising edge
4. n=2



Compare Match Output Mode – TnCCLR=1

Note: 1. With TnCCLR=1, A Comparator A match will clear the counter

2. The TM output pin is controlled only by the TnAF flag

3. The output pin is reset to its initial state by a TnON bit rising edge

4. A TnPF flag is not generated when TnCCLR=1

5. n=2

Timer/Counter Mode

To select this mode, bits TnM1 and TnM0 in the TMnC1 register should be set to 11 respectively. The Timer/Counter Mode operates in an identical way to the Compare Match Output Mode generating the same interrupt flags. The exception is that in the Timer/Counter Mode the TM output pin is not used. Therefore the above description and Timing Diagrams for the Compare Match Output Mode can be used to understand its function. As the TM output pin is not used in this mode, the pin can be used as a normal I/O pin or other pin-shared function.

PWM Output Mode

To select this mode, bits TnM1 and TnM0 in the TMnC1 register should be set to 10 respectively. The PWM function within the TM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the TM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM mode, the TnCCLR bit has no effect on the PWM operation. Both of the CCRA and CCRP registers are used to generate the PWM waveform, one register is used to clear the internal counter and thus control the PWM waveform frequency, while the other one is used to control the duty cycle. Which register is used to control either frequency or duty cycle is determined using the TnDPX bit in the TMnC1 register. The PWM waveform frequency and duty cycle can therefore be controlled by the values in the CCRA and CCRP registers.

An interrupt flag, one for each of the CCRA and CCRP, will be generated when a compare match occurs from either Comparator A or Comparator P. The TnOC bit in the TMnC1 register is used to select the required polarity of the PWM waveform while the two TnIO1 and TnIO0 bits are used to enable the PWM output or to force the TM output pin to a fixed high or low level. The TnPOL bit is used to reverse the polarity of the PWM output waveform.

16-bit STM, PWM Mode, Edge-aligned Mode, TnDPX=0

CCRP	1~255	0
Period	CCRP × 256	65536
Duty	CCRA	

If f_{SYS}=16MHz, TM clock source is f_{SYS}/4, CCRP=2 and CCRA=128

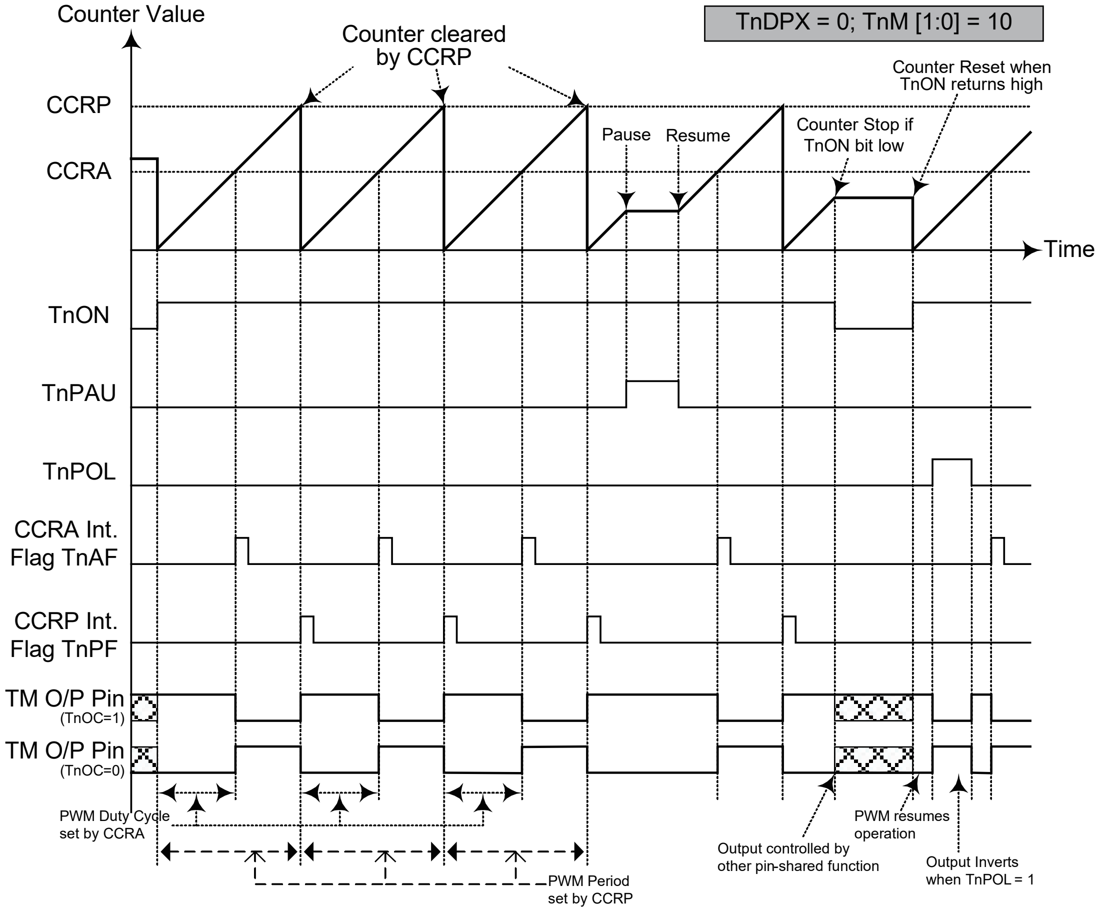
The STM PWM output frequency=(f_{SYS}/4)/(2×256)=f_{SYS}/2048=7.8125kHz, duty=128/(2×256)=25%.

If the Duty value defined by the CCRA register is equal to or greater than the Period value, then the PWM output duty is 100%.

16-bit STM, PWM Mode, Edge-aligned Mode, TnDPX=1

CCRP	1~255	0
Period	CCRA	
Duty	CCRP × 256	65536

The PWM output period is determined by the CCRA register value together with the TM clock while the PWM duty cycle is defined by the (CCRP×256) except when the CCRP value is equal to 0.



PWM Mode – TnDPX=0

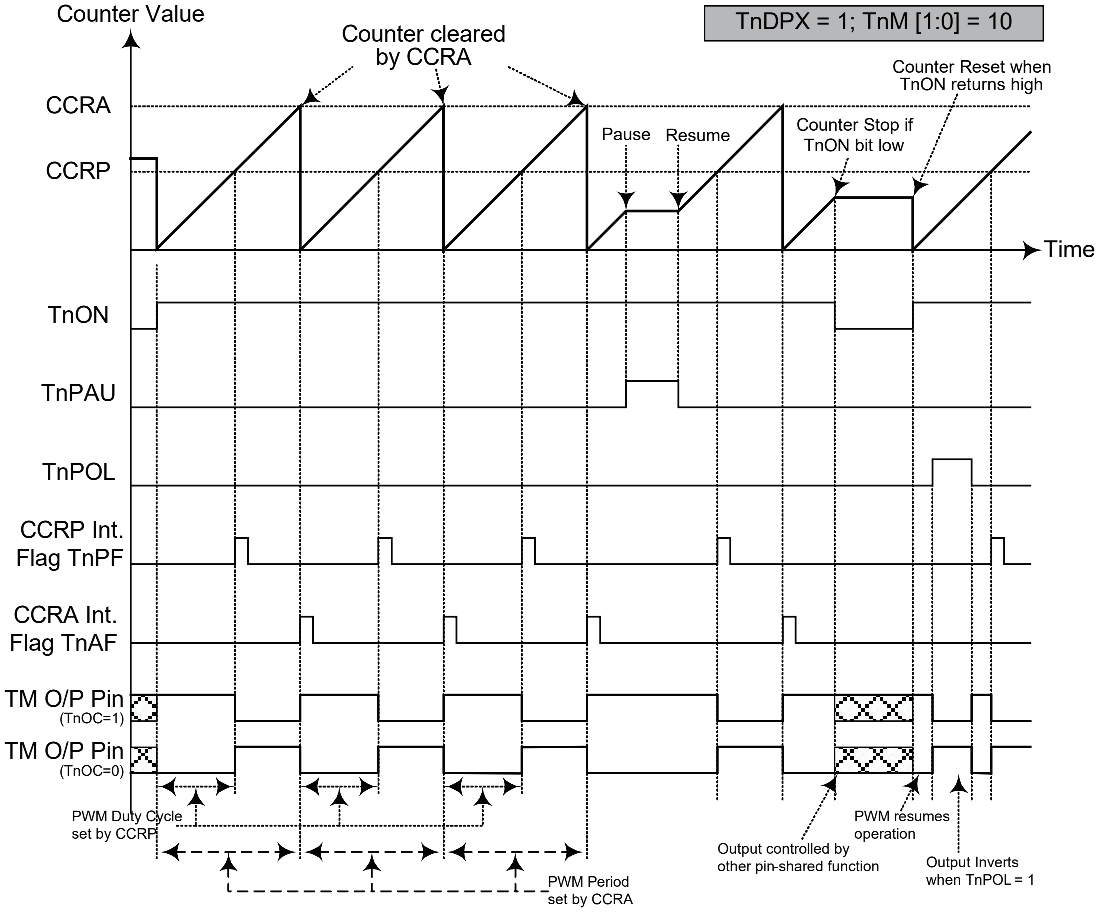
Note: 1. Here TnDPX=0 -- Counter cleared by CCRP

2. A counter clear sets the PWM Period

3. The internal PWM function continues running even when TnIO [1:0]=00 or 01

4. The TnCCLR bit has no influence on PWM operation

5. n=2



PWM Mode – TnDPX=1

Note: 1. Here TnDPX=1 -- Counter cleared by CCRA

2. A counter clear sets the PWM Period

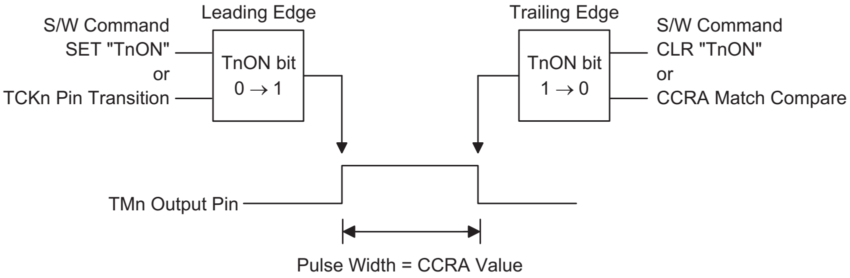
3. The internal PWM function continues running even when TnIO [1:0]=00 or 01

4. The TnCCLR bit has no influence on PWM operation

5. n=2

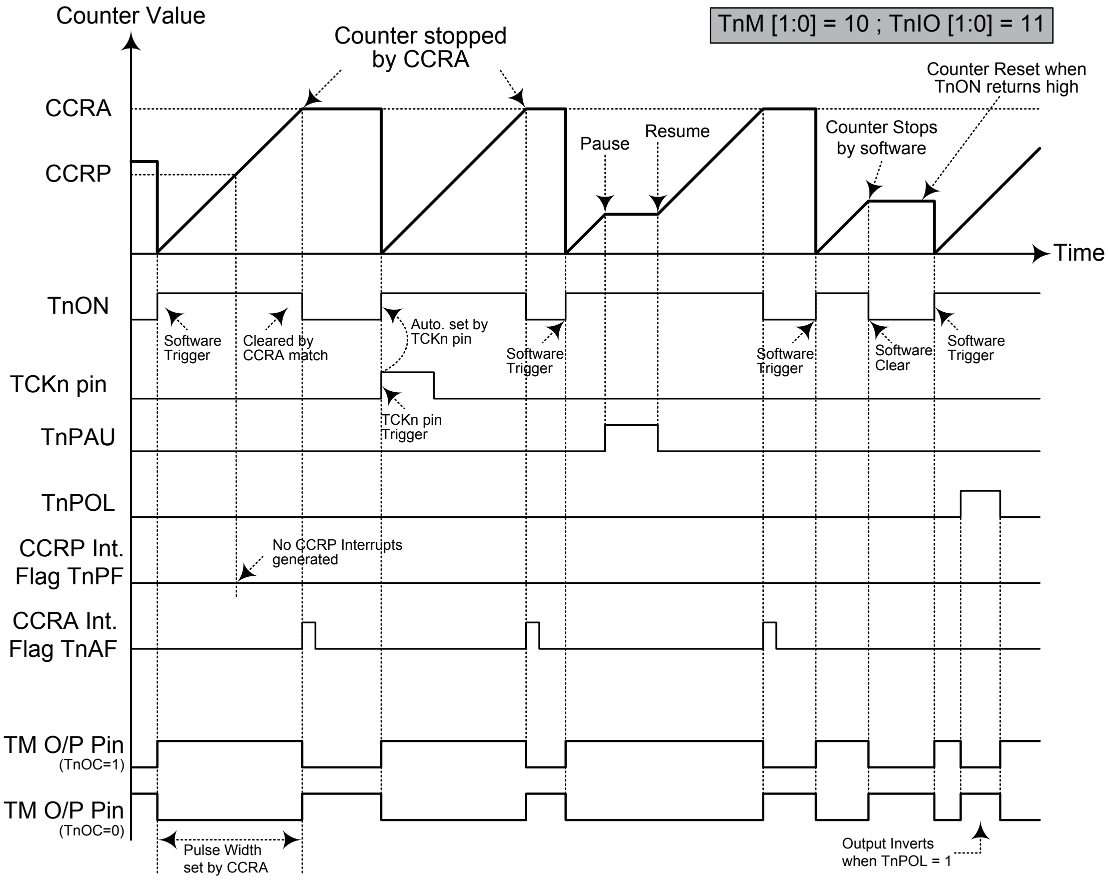
Single Pulse Mode

To select this mode, bits TnM1 and TnM0 in the TMnC1 register should be set to 10 respectively and also the TnIO1 and TnIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the TM output pin. The trigger for the pulse output leading edge is a low to high transition of the TnON bit, which can be implemented using the application program. However in the Single Pulse Mode, the TnON bit can also be made to automatically change from low to high using the external TCKn pin, which will in turn initiate the Single Pulse output. When the TnON bit transitions to a high level, the counter will start running and the pulse leading edge will be generated. The TnON bit should remain high when the pulse is in its active state. The generated pulse trailing edge will be generated when the TnON bit is cleared to zero, which can be implemented using the application program or when a compare match occurs from Comparator A.



Single Pulse Generation

However a compare match from Comparator A will also automatically clear the TnON bit and thus generate the Single Pulse output trailing edge. In this way the CCRA value can be used to control the pulse width. A compare match from Comparator A will also generate a TM interrupt. The counter can only be reset back to zero when the TnON bit changes from low to high when the counter restarts. In the Single Pulse Mode CCRP is not used. The TnCCLR and TnDPX bits are not used in this Mode.



Single Pulse Mode

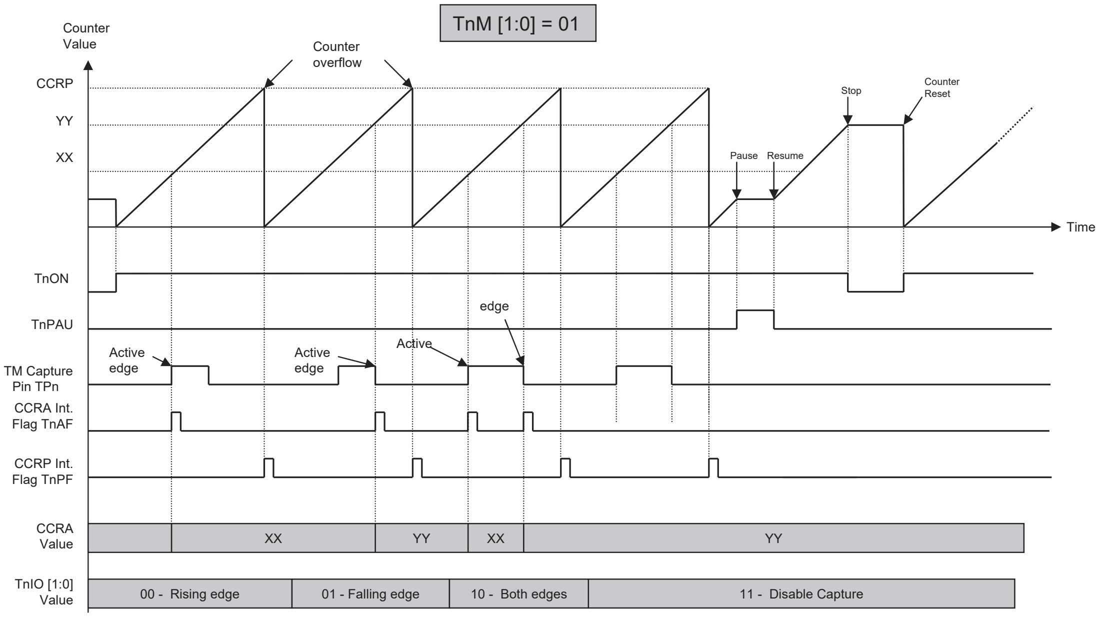
Note: 1. Counter stopped by CCRA
2. CCRP is not used
3. The pulse triggered by the TCKn pin or by setting the TnON bit high
4. A TCKn pin active edge will automatically set the TnON bit high
5. In the Single Pulse Mode, TnIO [1:0] must be set to 11 and can not be changed
6. n=2

Capture Input Mode

To select this mode bits TnM1 and TnM0 in the TMnC1 register should be set to 01 respectively. This mode enables external signals to capture and store the present value of the internal counter and can therefore be used for applications such as pulse width measurements. The external signal is supplied on the TPn pin, whose active edge can be either a rising edge, a falling edge or both rising and falling edges; the active edge transition type is selected using the TnIO1 and TnIO0 bits in the TMnC1 register. The counter is started when the TnON bit changes from low to high which is initiated using the application program.

When the required edge transition appears on the TPn pin, the present value in the counter will be latched into the CCRA registers and a TM interrupt generated. Irrespective of what events occur on the TPn pin the counter will continue to free run until the TnON bit changes from high to low. When a CCRP compare match occurs, the counter will reset back to zero; in this way the CCRP value can be used to control the maximum counter value. When a CCRP compare match occurs from Comparator P, a TM interrupt will also be generated. Counting the number of overflow interrupt signals from the CCRP can be a useful method in measuring long pulse widths. The TnIO1 and TnIO0 bits can select the active trigger edge on the TPn pin to be a rising edge, falling edge or both edge types. If the TnIO1 and TnIO0 bits are both set high, then no capture operation will take place irrespective of what happens on the TPn pin, however it must be noted that the counter will continue to run.

As the TPn pin is pin shared with other functions, care must be taken if the TM is in the Input Capture Mode. This is because if the pin is setup as an output, then any transitions on this pin may cause an input capture operation to be executed. The TnCCLR and TnDPX bits are not used in this Mode.



Capture Input Mode

Note: 1. TnM [1:0]=01 and active edge set by the TnIO [1:0] bits

2. A TM Capture input pin active edge transfers the counter value to CCRA

3. The TnCCLR bit is not used

4. No output function -- TnOC and TnPOL bits are not used

5. CCRP determines the counter value and the counter has a maximum count value when CCRP is equal to zero

6. n=2

Enhanced Type TM – ETM

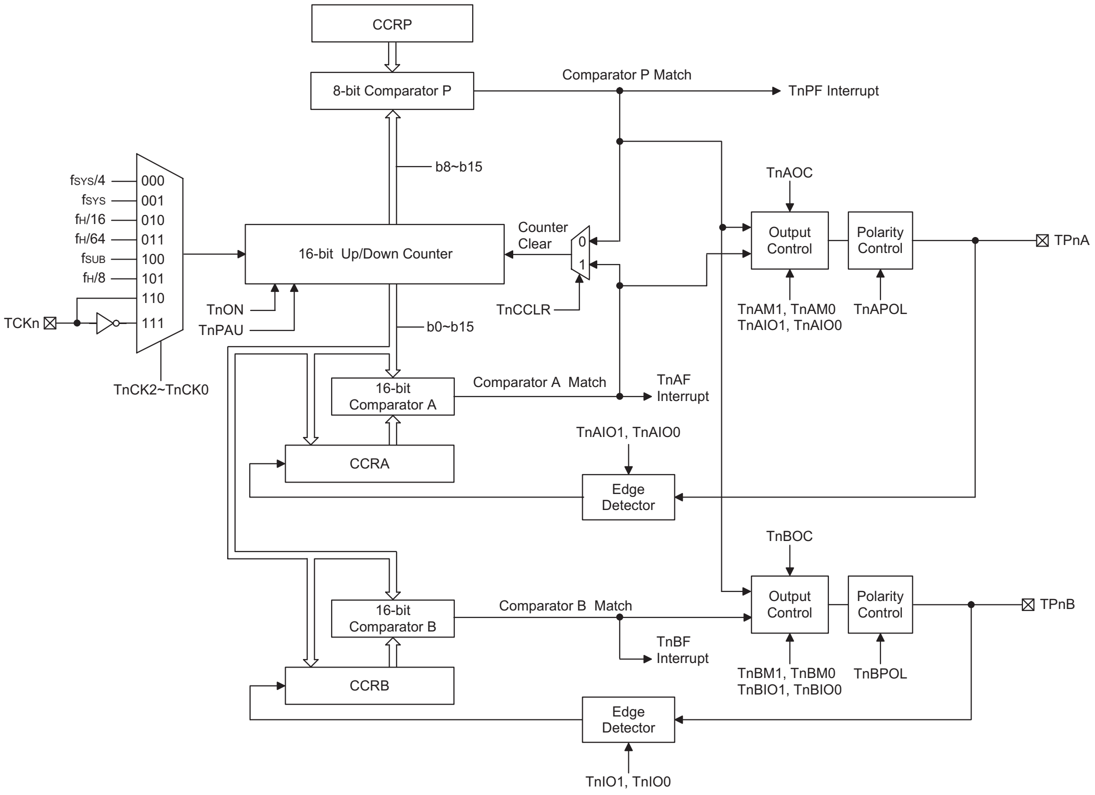
The Enhanced Type TM contains five operating modes, which are Compare Match Output, Timer/Event Counter, Capture Input, Single Pulse Output and PWM Output modes. The Enhanced TM can also be controlled with an external input pin and can drive three or four external output pins.

ETM	Name	TM No.	TM Input Pin	TM Output Pin
BC66F840 BC66F850 BC66F860	16-bit ETM	3	TCK3	TP3

Enhanced TM Operation

At its core is a 16-bit count-up/count-down counter which is driven by a user selectable internal or external clock source. There are three internal comparators with the names, Comparator A, Comparator B and Comparator P. These comparators will compare the value in the counter with the CCRA, CCRB and CCRP registers. The CCRP comparator is 8 bits wide whose value is compared with the highest 8 bits in the counter while CCRA and CCRB are 16 bits wide and therefore compared with all counter bits.

The only way of changing the value of the 16-bit counter using the application program, is to clear the counter by changing the TnON bit from low to high. The counter will also be cleared automatically by a counter overflow or a compare match with one of its associated comparators. When these conditions occur, a TM interrupt signal will also usually be generated. The Enhanced Type TM can operate in a number of different operational modes, can be driven by different clock sources including an input pin and can also control output pins. All operating setup conditions are selected using relevant internal registers.



Enhanced Type TM Block Diagram (n=3)

Enhanced Type TM Register Description

Overall operation of the Enhanced TM is controlled using a series of registers. A read only register pair exists to store the internal counter 16-bit value, while two read/write register pairs exist to store the internal 16-bit CCRA and CCRB value. The remaining three registers are control registers which setup the different operating and control modes as well as the eight CCRP bits.

Register Name	Bit							
	7	6	5	4	3	2	1	0
TMnC0	TnPAU	TnCK2	TnCK1	TnCK0	TnON	—	—	—
TMnC1	TnAM1	TnAM0	TnAIO1	TnAIO0	TnAOC	TnAPOL	TnCDN	TnCCLR
TMnC2	TnBM1	TnBM0	TnBIO1	TnBIO0	TnBOC	TnBPOL	TnPWM1	TnPWM0
TMnDL	D7	D6	D5	D4	D3	D2	D1	D0
TMnDH	D15	D14	D13	D12	D11	D10	D9	D8
TMnAL	D7	D6	D5	D4	D3	D2	D1	D0
TMnAH	D15	D14	D13	D12	D11	D10	D9	D8
TMnBL	D7	D6	D5	D4	D3	D2	D1	D0
TMnBH	D15	D14	D13	D12	D11	D10	D9	D8
TMnRP	TnRP7	TnRP6	TnRP5	TnRP4	TnRP3	TnRP2	TnRP1	TnRP0

16-bit Enhanced TM Register List (n=3)

TMnDL Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TMnDL:** TMn Counter Low Byte Register bit 7~bit 0
TMn 16-bit Counter bit 7~bit 0

TMnDH Register

Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TMnDH:** TMn Counter High Byte Register bit 7~bit 0
TMn 16-bit Counter bit 15~bit 8

TMnAL Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TMnAL:** TMn CCRA Low Byte Register bit 7~bit 0
TMn 16-bit CCRA bit 7~bit 0

TMnAH Register

Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TMnAH:** TMn CCRA High Byte Register bit 7~bit 0
TMn 16-bit CCRA bit 15~bit 8

TMnBL Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TMnBL:** TMn CCRB Low Byte Register bit 7~bit 0
TMn 16-bit CCRB bit 7~bit 0

TMnBH Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~0 **TMnBH:** TMn CCRB High Byte Register bit 7~bit 0
TMn 16-bit CCRB bit 15~bit 8

TMnC0 Register

Bit	7	6	5	4	3	2	1	0
Name	TnPAU	TnCK2	TnCK1	TnCK0	TnON	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

Bit 7 **TnPAU:** TMn Counter Pause Control
0: Run
1: Pause

The counter can be paused by setting this bit high. Clearing the bit to zero restores normal counter operation. When in a Pause condition the TM will remain powered up and continue to consume power. The counter will retain its residual value when this bit changes from low to high and resume counting from this value when the bit changes to a low value again.

Bit 6~4 **TnCK2~TnCK0:** Select TMn Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: $f_H/8$
110: TCKn rising edge clock
111: TCKn falling edge clock

These three bits are used to select the clock source for the TM. Selecting the Reserved clock input will effectively disable the internal counter. The external pin clock source can be chosen to be active on the rising or falling edge. The clock source f_{SYS} is the system clock, while f_H and f_{SUB} are other internal clocks, the details of which can be found in the oscillator section.

Bit 3 **TnON:** TMn Counter On/Off Control
0: Off
1: On

This bit controls the overall on/off function of the TM. Setting the bit high enables the counter to run while clearing the bit disables the TM. Clearing this bit to zero will stop the counter from counting and turn off the TM which will reduce its power consumption. When the bit changes state from low to high the internal counter value will be reset to zero, however when the bit changes from high to low, the internal counter will retain its residual value until the bit returns high again.

If the TM is in the Compare Match Output Mode, then the TM output pin will be reset to its initial condition, as specified by the TnOC bit, when the TnON bit changes from low to high.

Bit 7~2 Unimplemented, read as “0”

TMnC1 Register

Bit	7	6	5	4	3	2	1	0
Name	TnAM1	TnAM0	TnAIO1	TnAIO0	TnAOC	TnAPOL	TnCDN	TnCCLR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **TnAM1~TnAM0:** Select TMn CCRA Operation Mode
00: Compare Match Output Mode
01: Capture Input Mode
10: PWM Mode or Single Pulse Output Mode
11: Timer/Counter Mode

These bits setup the required operating mode for the TM. To ensure reliable operation the TM should be switched off before any changes are made to the TnAM1 and TnAM0 bits. In the Timer/Counter Mode, the TM output pin control must be disabled.

Bit 5~4 **TnAIO1~TnAIO0:** Select TPnA output function
Compare Match Output Mode
00: No change
01: Output low
10: Output high
11: Toggle output
PWM Mode/Single Pulse Output Mode
00: PWM Output inactive state
01: PWM Output active state
10: PWM output
11: Single pulse output
Capture input Mode
00: Input capture at rising edge of TPnA
01: Input capture at falling edge of TPnA
01: Input capture at falling/rising edge of TPnA
11: Input capture disabled
Timer/Counter Mode
Unused

These two bits are used to determine how the TM output pin changes state when a certain condition is reached. The function that these bits select depends upon in which mode the TM is running.

In the Compare Match Output Mode, the TnAIO1 and TnAIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the TnAOC bit in the TMnC1 register. Note that the output level requested by the TnAIO1 and TnAIO0 bits must be different from the initial value setup using the TnAOC bit otherwise no change will occur on the TM output pin when a compare match occurs. After the TM output pin changes state, it can be reset to its initial level by changing the level of the TnON bit from low to high.

In the PWM Mode, the TnAIO1 and TnAIO0 bits determine how the TM output pin changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to change the values of the TnAIO1 and TnAIO0 bits only after the TM has been switched off. Unpredictable PWM outputs will occur if the TnAIO1 and TnAIO0 bits are changed when the TM is running.

Bit 3 **TnAOC:** TPnA output control bit
Compare Match Output Mode
0: Initial low
1: Initial high
PWM Mode/Single Pulse Output Mode
0: Active low
1: Active high

This is the output control bit for the TM output pin. Its operation depends upon whether TM is being used in the Compare Match Output Mode or in the PWM Mode/ Single Pulse Output Mode. It has no effect if the TM is in the Timer/Counter Mode. In the Compare Match Output Mode it determines the logic level of the TM output pin before a compare match occurs. In the PWM Mode it determines if the PWM signal is active high or active low.

Bit 2 **TnAPOL:** TPnA output Polarity control
0: Non-invert
1: Invert

This bit controls the polarity of the TPnA output pin. When the bit is set high the TM output pin will be inverted and not inverted when the bit is zero. It has no effect if the TM is in the Timer/Counter Mode.

Bit 1 **TnCDN:** TMn Counter count up or down flag
0: Count Up
1: Count Down

Bit 0 **TnCCLR:** Select TMn Counter clear condition
0: TMn Comparator P match
1: TMn Comparator A match

This bit is used to select the method which clears the counter. Remember that the Enhanced TM contains three comparators, Comparator A, Comparator B and Comparator P, but only Comparator A or Comparator P can be selected to clear the internal counter. With the TnCCLR bit set high, the counter will be cleared when a compare match occurs from the Comparator A. When the bit is low, the counter will be cleared when a compare match occurs from the Comparator P or with a counter overflow. A counter overflow clearing method can only be implemented if the CCRP bits are all cleared to zero. The TnCCLR bit is not used in the Single Pulse or Input Capture Mode.

TMnC2 Register

Bit	7	6	5	4	3	2	1	0
Name	TnBM1	TnBM0	TnBIO1	TnBIO0	TnBOC	TnBPOL	TnPWM1	TnPWM0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **TnBM1~TnBM0:** Select TMn CCRB Operation Mode
00: Compare Match Output Mode
01: Capture Input Mode
10: PWM Mode or Single Pulse Output Mode
11: Timer/Counter Mode

These bits setup the required operating mode for the TM. To ensure reliable operation the TM should be switched off before any changes are made to the TnBM1 and TnBM0 bits. In the Timer/Counter Mode, the TM output pin control must be disabled.

Bit 5~4 **TnBIO1~TnBIO0:** Select TPnB output function
Compare Match Output Mode
00: No change
01: Output low
10: Output high
11: Toggle output
PWM Mode/Single Pulse Output Mode
00: PWM Output inactive state
01: PWM Output active state
10: PWM output
11: Single pulse output
Capture input Mode
00: Input capture at rising edge of TPnB
01: Input capture at falling edge of TPnB
01: Input capture at falling/rising edge of TPnB
11: Input capture disabled
Timer/Counter Mode
Unused

These two bits are used to determine how the TM output pin changes state when a certain condition is reached. The function that these bits select depends upon in which mode the TM is running.

In the Compare Match Output Mode, the TnBIO1 and TnBIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the TnBOC bit in the TMnC2 register. Note that the output level requested by the TnBIO1 and TnBIO0 bits must be different from the initial value setup using the TnBOC bit otherwise no change will occur on the TM output pin when a compare match occurs. After the TM output pin changes state, it can be reset to its initial level by changing the level of the TnON bit from low to high.

In the PWM Mode, the TnBIO1 and TnBIO0 bits determine how the TM output pin changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to change the values of the TnBIO1 and TnBIO0 bits only after the TM has been switched off. Unpredictable PWM outputs will occur if the TnBIO1 and TnBIO0 bits are changed when the TM is running.

Bit 3 **TnBOC:** TPnB output control bit

Compare Match Output Mode
0: Initial low
1: Initial high

PWM Mode/Single Pulse Output Mode
0: Active low
1: Active high

This is the output control bit for the TM output pin. Its operation depends upon whether TM is being used in the Compare Match Output Mode or in the PWM Mode/Single Pulse Output Mode. It has no effect if the TM is in the Timer/Counter Mode. In the Compare Match Output Mode it determines the logic level of the TM output pin before a compare match occurs. In the PWM Mode it determines if the PWM signal is active high or active low.

Bit 2 **TnBPOL:** TPnB output Polarity control
0: Non-invert
1: Invert

This bit controls the polarity of the TPnB output pin. When the bit is set high the TM output pin will be inverted and not inverted when the bit is zero. It has no effect if the TM is in the Timer/Counter Mode.

Bit 1~0 **TnPWM1~TnPWM0:** Select PWM Mode
00: Edge aligned
01: Centre aligned, compare match on count up
10: Centre aligned, compare match on count down
11: Centre aligned, compare match on count up or down

TMnRP Register

Bit	7	6	5	4	3	2	1	0
Name	TnRP7	TnRP6	TnRP5	TnRP4	TnRP3	TnRP2	TnRP1	TnRP0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **TnRP7~TnRP0:** TMn CCRP register bit 7~bit 0, compare with the TMn counter bit 15~bit 8

Comparator P Match Period
0: 65536 TMn clocks
1~255: (1~255)×256 TMn clocks

These eight bits are used to setup the value on the internal CCRP 8-bit register, which are then compared with the internal counter's highest eight bits. The result of this comparison can be selected to clear the internal counter if the TnCCLR bit is set to zero. Setting the TnCCLR bit to zero ensures that a compare match with the CCRP values will reset the internal counter. As the CCRP bits are only compared with the highest eight counter bits, the compare values exist in 256 clock cycle multiples. Clearing all eight bits to zero is in effect allowing the counter to overflow at its maximum value.

Enhanced Type TM Operation Modes

The Enhanced Type TM can operate in one of five operating modes, Compare Match Output Mode, PWM Output Mode, Single Pulse Output Mode, Capture Input Mode or Timer/Counter Mode. The operating mode is selected using the TnAM1 and TnAM0 bits in the TMnC1, and the TnBM1 and TnBM0 bits in the TMnC2 register.

ETM Operating Mode	CCRA Compare Match Output Mode	CCRA Timer/ Counter Mode	CCRA PWM Output Mode	CCRA Single Pulse Output Mode	CCRA Input Capture Mode
CCRB Compare Match Output Mode	√	—	—	—	—
CCRB Timer/Counter Mode	—	√	—	—	—
CCRB PWM Output Mode	—	—	√	—	—
CCRB Single Pulse Output Mode	—	—	—	√	—
CCRB Input Capture Mode	—	—	—	—	√

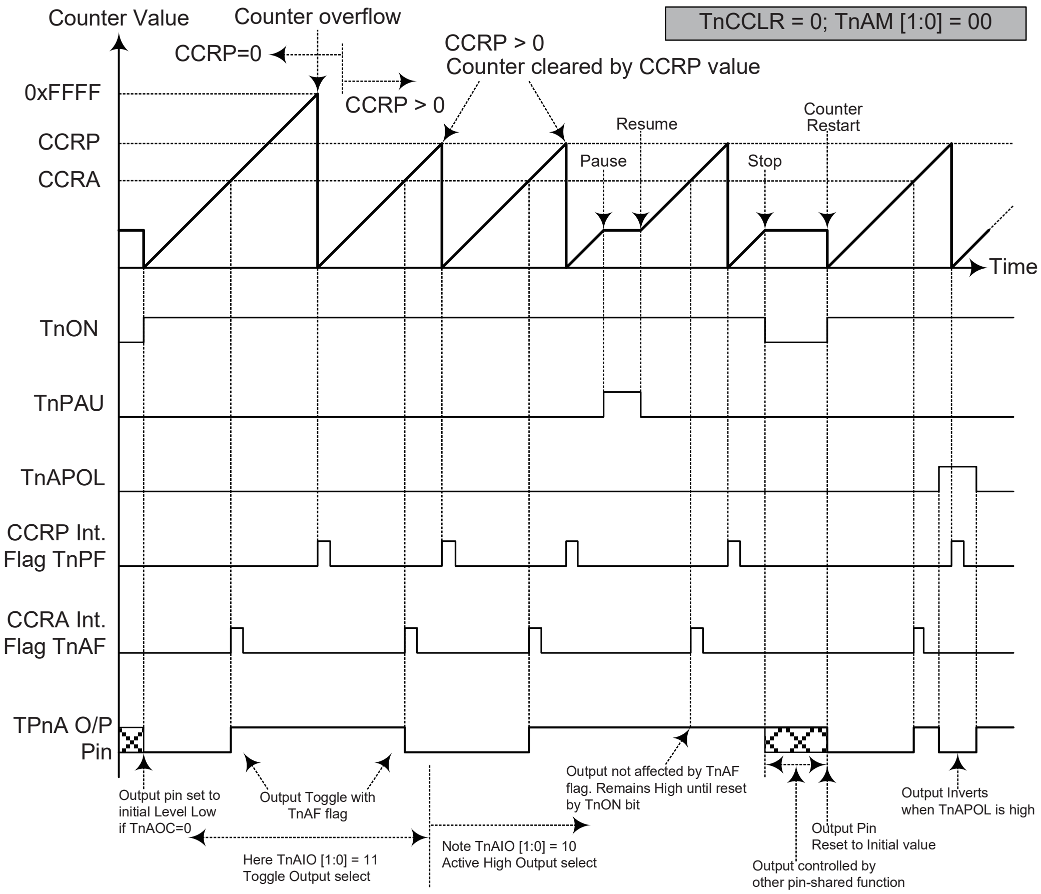
"√": permitted, "—": not permitted

Compare Match Output Mode

To select this mode, bits TnAM1, TnAM0 and TnBM1, TnBM0 in the TMnC1/TMnC2 registers should be all cleared to zero. In this mode once the counter is enabled and running it can be cleared by three methods. These are a counter overflow, a compare match from Comparator A and a compare match from Comparator P. When the TnCCLR bit is low, there are two ways in which the counter can be cleared. One is when a compare match occurs from Comparator P, the other is when the CCRP bits are all zero which allows the counter to overflow. Here both the TnAF and TnPF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

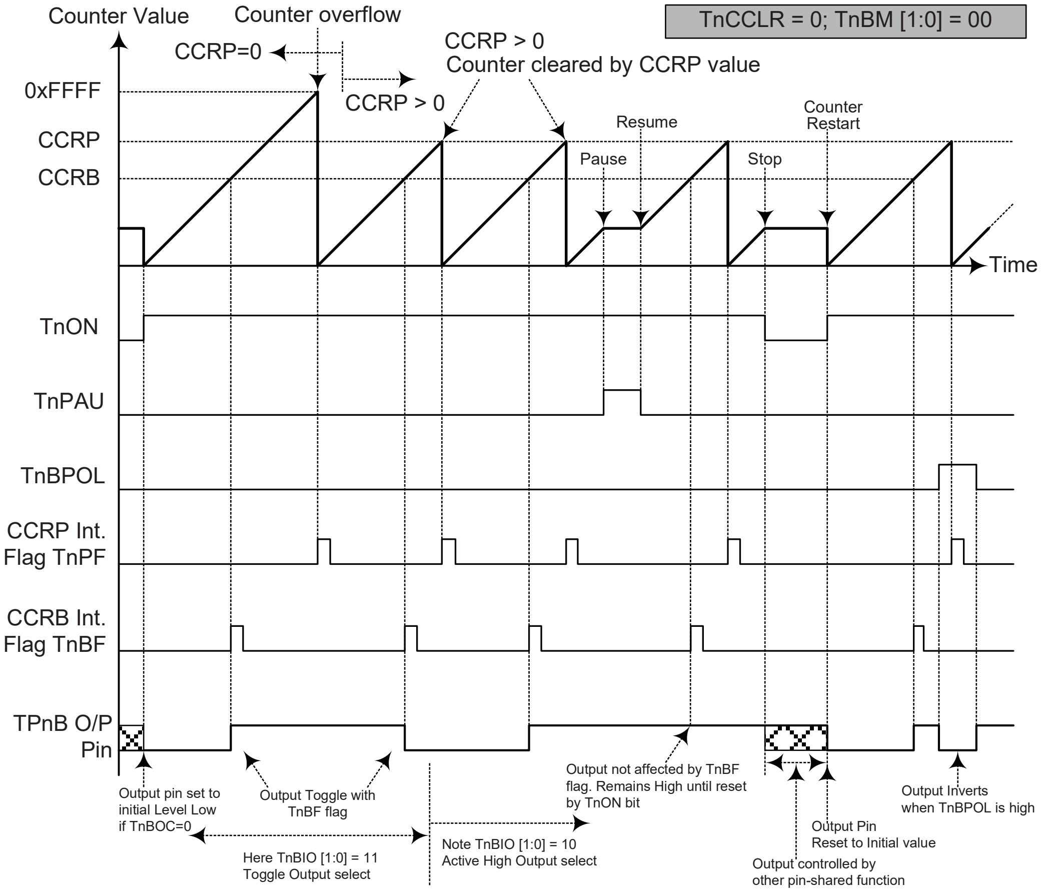
If the TnCCLR bit in the TMnC1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the TnAF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when TnCCLR is high no TnPF interrupt request flag will be generated.

As the name of the mode suggests, after a comparison is made, the TM output pin, will change state. The TM output pin condition however only changes state when a TnAF or TnBF interrupt request flag is generated after a compare match occurs from Comparator A or Comparator B. The TnPF interrupt request flag, generated from a compare match from Comparator P, will have no effect on the TM output pin. The way in which the TM output pin changes state is determined by the condition of the TnAIO1 and TnAIO0 bits in the TMnC1 register for ETM CCRA, and the TnBIO1 and TnBIO0 bits in the TMnC2 register for ETM CCRB. The TM output pin can be selected using the TnAIO1, TnAIO0 bits for the TPnA pin and TnBIO1, TnBIO0 bits for the TPnB pin to go high, to go low or to toggle from its present condition when a compare match occurs from Comparator A or a compare match occurs from Comparator B. The initial condition of the TM output pin, which is setup after the TnON bit changes from low to high, is setup using the TnAOC or TnBOC bit for TPnA or TPnB output pin. Note that if the TnAIO1, TnAIO0 and TnBIO1, TnBIO0 bits are zero then no pin change will take place.



ETM CCRA Compare Match Output Mode – TnCCLR=0

Note: 1. With TnCCLR=0, a Comparator P match will clear the counter
2. The TPnA output pin is controlled only by the TnAF flag
3. The output pin is reset to its initial state by a TnON bit rising edge
4. n=3



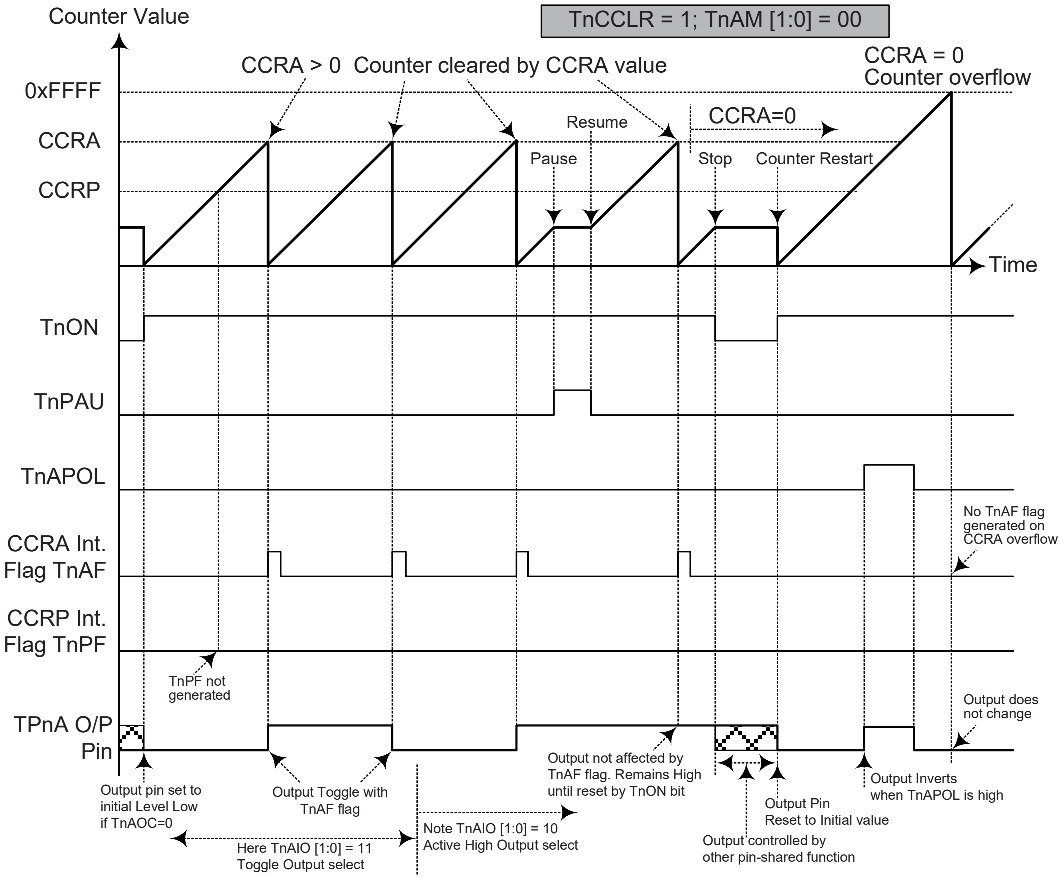
ETM CCRB Compare Match Output Mode – TnCCLR=0

Note: 1. With TnCCLR=0, a Comparator P match will clear the counter

2. The TPnB output pin is controlled only by the TnBF flag

3. The output pin is reset to its initial state by a TnON bit rising edge

4. n=3


Counter Value
TnCCLR = 1; TnAM [1:0] = 00
CCRA > 0 Counter cleared by CCRA value
CCRA = 0
Counter overflow
0xFFFF
CCRA
CCRP
Resume
CCRA=0
Pause
Stop
Counter Restart
Time
TnON
TnPAU
TnAPOL
CCRA Int. Flag TnAF
No TnAF flag generated on CCRA overflow
CCRP Int. Flag TnPF
TnPF not generated
TPnA O/P Pin
Output does not change
Output pin set to initial Level Low if TnAOC=0
Output Toggle with TnAF flag
Output not affected by TnAF flag. Remains High until reset by TnON bit
Output Pin Reset to Initial value
Output Inverts when TnAPOL is high
Here TnAIO [1:0] = 11 Toggle Output select
Note TnAIO [1:0] = 10 Active High Output select
Output controlled by other pin-shared function

ETM CCRA Compare Match Output Mode – TnCCLR=1

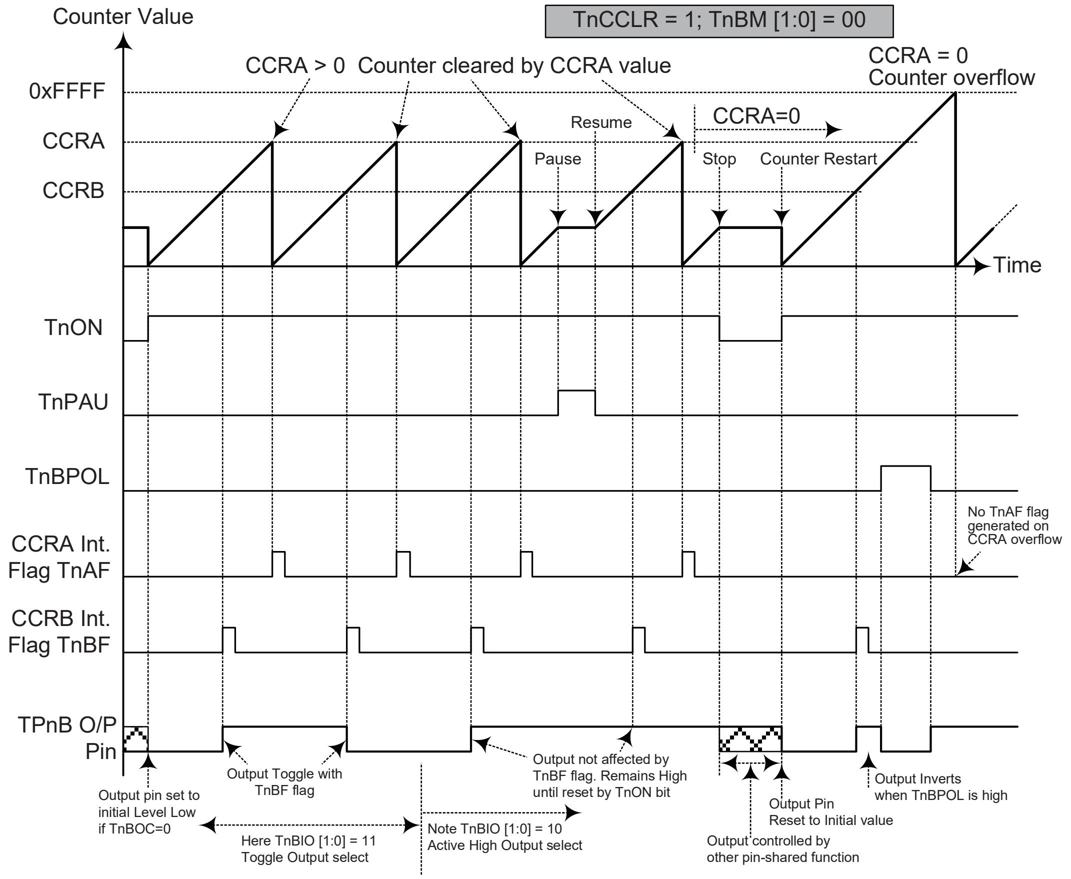
Note: 1. With TnCCLR=1, a Comparator A match will clear the counter

2. The TPnA output pin is controlled only by the TnAF flag

3. The TPnA output pin is reset to its initial state by a TnON bit rising edge

4. The TnPF flag is not generated when TnCCLR=1

5. n=3



ETM CCRB Compare Match Output Mode – TnCCLR=1

Note: 1. With TnCCLR=1, a Comparator A match will clear the counter

2. The TPnB output pin is controlled only by the TnBF flag

3. The TPnB output pin is reset to its initial state by a TnON bit rising edge

4. The TnPF flag is not generated when TnCCLR=1

5. n=3

Timer/Counter Mode

To select this mode, bits TnAM1, TnAM0 and TnBM1, TnBM0 in the TMnC1 and TMnC2 register should all be set high. The Timer/Counter Mode operates in an identical way to the Compare Match Output Mode generating the same interrupt flags. The exception is that in the Timer/Counter Mode the TM output pin is not used. Therefore the above description and Timing Diagrams for the Compare Match Output Mode can be used to understand its function. As the TM output pin is not used in this mode, the pin can be used as a normal I/O pin or other pin-shared function.

PWM Output Mode

To select this mode, the required bit pairs, TnAM1, TnAM0 and TnBM1, TnBM0 should be set to 10 respectively and also the TnAIO1, TnAIO0 and TnBIO1, TnBIO0 bits should be set to 10 respectively. The PWM function within the TM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the TM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM mode, the TnCCLR bit is used to determine in which way the PWM period is controlled. With the TnCCLR bit set high, the PWM period can be finely controlled using the CCRA registers. In this case the CCRB registers are used to set the PWM duty value (for TPnB output pins). The CCRP bits are not used and TPnA output pin is not used. The PWM output can only be generated on the TPnB output pins. With the TnCCLR bit cleared to zero, the PWM period is set using the eight CCRP bits, in multiples of 256. Now both CCRA and CCRB registers can be used to setup different duty cycle values to provide dual PWM outputs on their relative TPnA and TPnB pins.

The TnPWM1 and TnPWM0 bits determine the PWM alignment type, which can be either edge or centre type. In edge alignment, the leading edge of the PWM signals will all be generated concurrently when the counter is reset to zero. With all power currents switching on at the same time, this may give rise to problems in higher power applications. In centre alignment the centre of the PWM active signals will occur sequentially, thus reducing the level of simultaneous power switching currents.

Interrupt flags, one for each of the CCRA, CCRB and CCRP, will be generated when a compare match occurs from either the Comparator A, Comparator B or Comparator P. The TnAOC and TnBOC bits in the TMnC1 and TMnC2 register are used to select the required polarity of the PWM waveform while the two TnAIO1, TnAIO0 and TnBIO1, TnBIO0 bits pairs are used to enable the PWM output or to force the TM output pin to a fixed high or low level. The TnAPOL and TnBPOL bit are used to reverse the polarity of the PWM output waveform.

ETM, PWM Mode, Edge-aligned Mode, TnCCLR=0

CCRP	1	2	…	127	128	…	254	255
Period	1×256	2×256	…	127×256	128×256	…	254×256	255×256
A Duty	CCRA							
B Duty	CCRB							

If f_{SYS}=16MHz, TM clock source select f_{SYS}/4, CCRP=2, CCRA=128 and CCRB=256,

The TP1A PWM output frequency=(f_{SYS}/4)/(2×256)=f_{SYS}/2048=7.8125kHz, duty=128/512=25%.

The TP1B PWM output frequency=(f_{SYS}/4)/(2×256)=f_{SYS}/2048=7.8125kHz, duty=256/512=50%.

If the Duty value defined by CCRA or CCRB register is equal to or greater than the Period value, then the PWM output duty is 100%.

ETM, PWM Mode, Edge-aligned Mode, TnCCLR=1

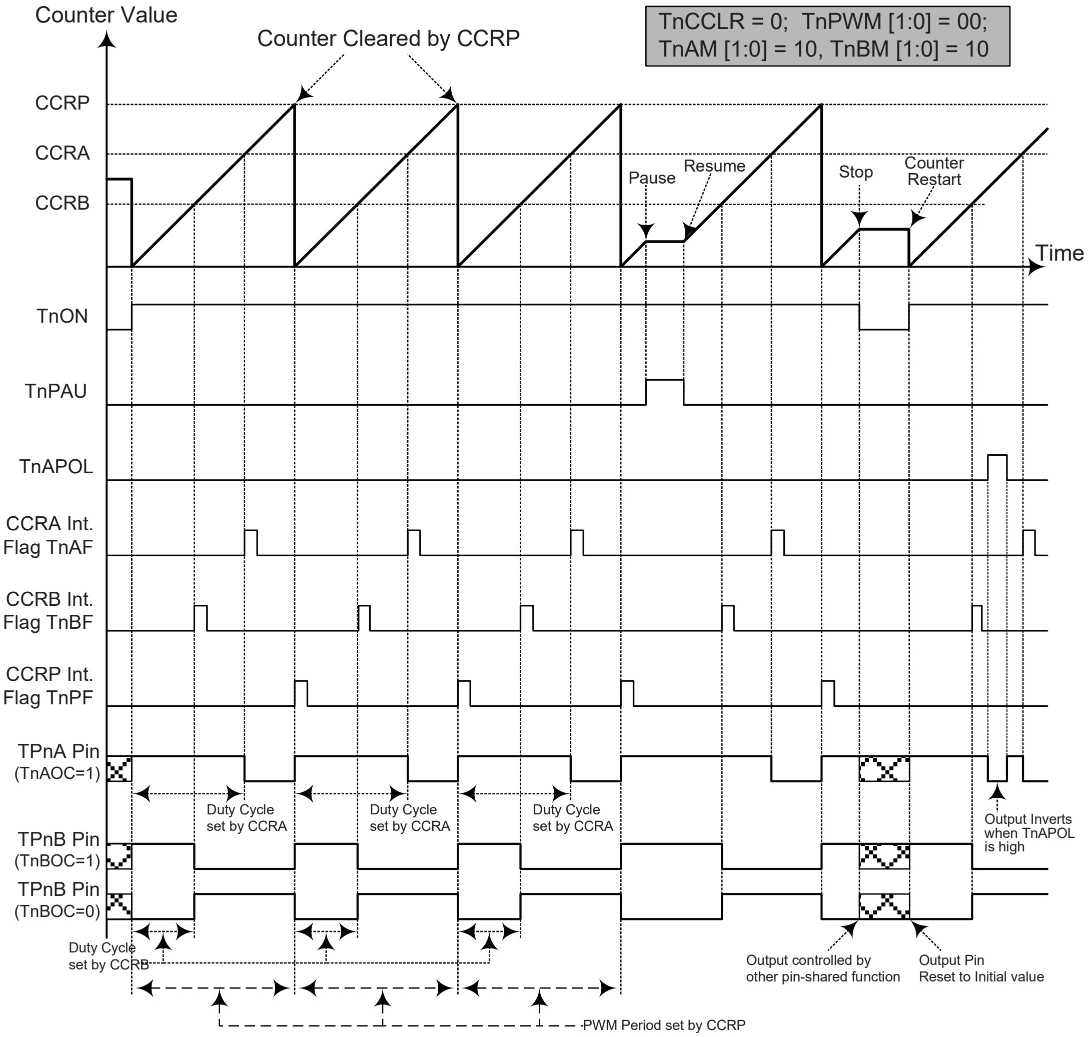
CCRA	1	2	…	32767	32768	…	65534	65535
Period	1	2	…	32767	32768	…	65534	65535
B Duty	CCRB							

ETM, PWM Mode, Center-aligned Mode, TnCCLR=0

CCRP	1	2	…	127	128	…	254	255
Period	1×256	2×256	…	127×256	128×256	…	254×256	255×256
A Duty	(CCRA × 2) - 1							
B Duty	(CCRB × 2) - 1							

ETM, PWM Mode, Center-aligned Mode, TnCCLR=1

CCRA	1	2	…	32767	32768	…	65534	65535
Period	2	4	…	65534	65536	…	131068	131070
B Duty	(CCRB × 2) - 1							



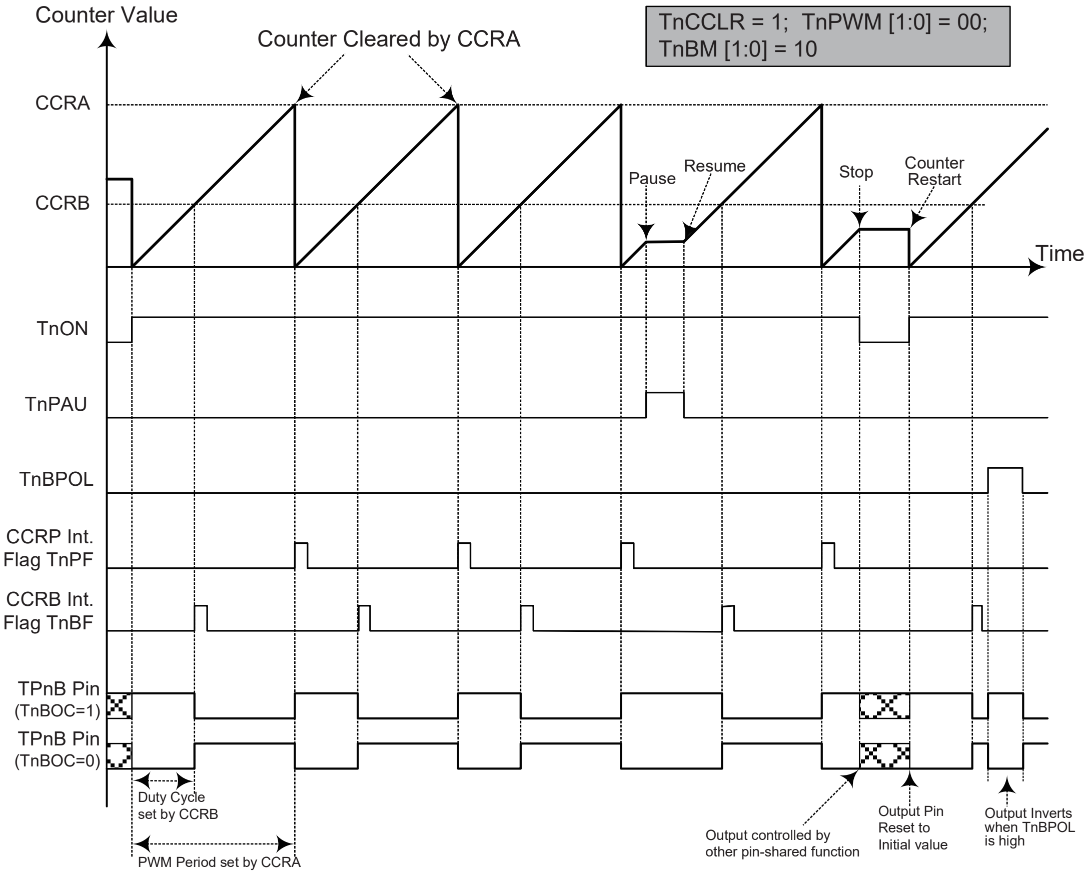
ETM PWM Mode – Edge Aligned

Note: 1. Here TnCCLR=0 therefore CCRP clears the counter and determines the PWM period

2. The internal PWM function continues running even when TnAIO [1:0] (or TnBIO [1:0])=00 or 01

3. CCRA controls the TPnA PWM duty and CCRB controls the TPnB PWM duty

4. n=3



ETM PWM Mode – Edge Aligned

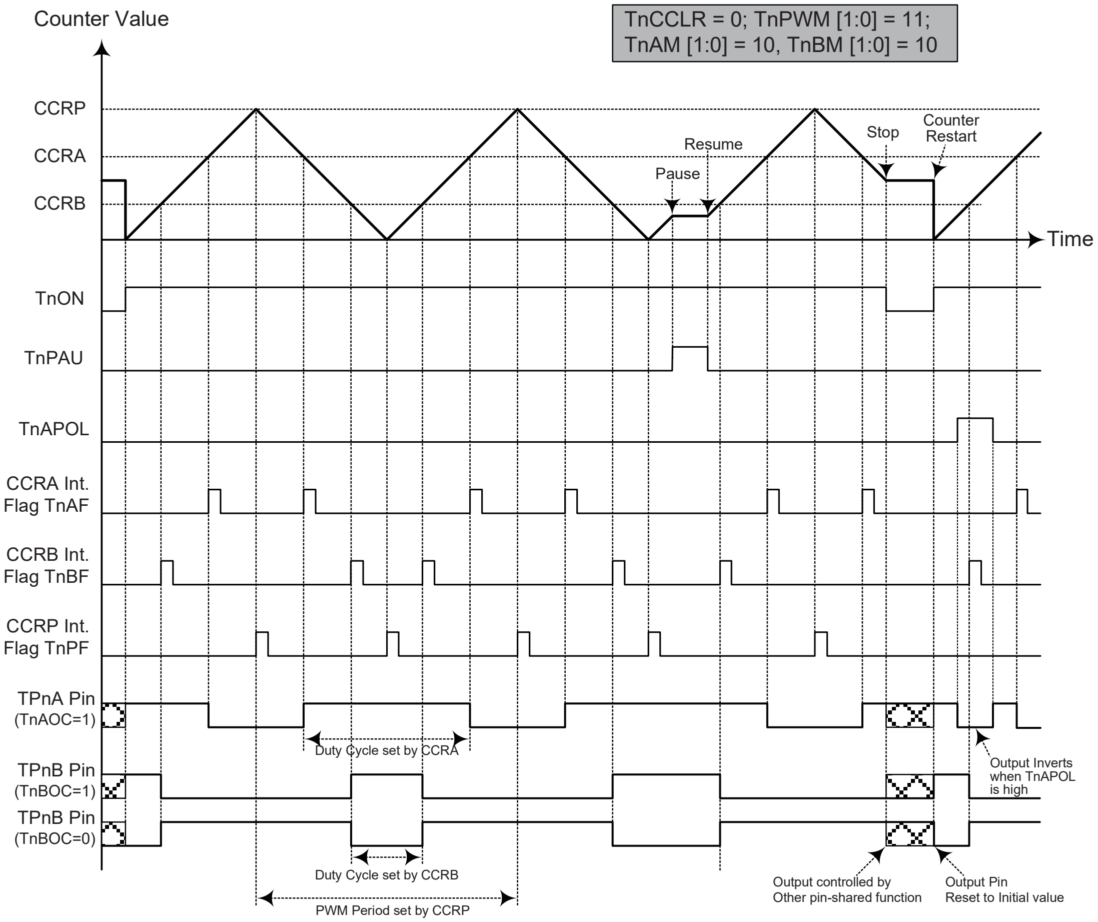
Note: 1. Here TnCCLR=1 therefore CCRA clears the counter and determines the PWM period

2. The internal PWM function continues running even when TnBIO [1:0]=00 or 01

3. CCRA controls the TPnB PWM period and CCRB controls the TPnB PWM duty

4. Here the TM pin control register should not enable the TPnA pin as a TM output pin

5. n=3



ETM PWM Mode – Centre Aligned

Note: 1. Here TnCCLR=0 therefore CCRP clears the counter and determines the PWM period

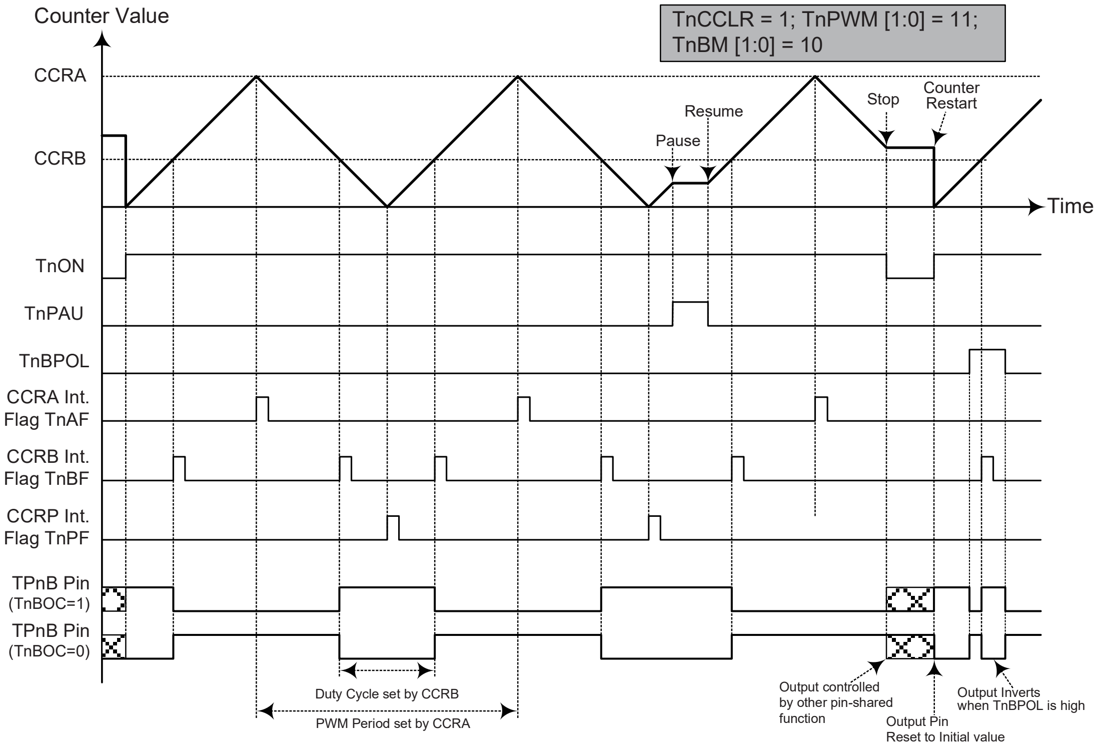
2. TnPWM [1:0]=11 therefore the PWM is centre aligned

3. The internal PWM function continues running even when TnAIO [1:0] (or TnBIO [1:0])=00 or 01

4. CCRA controls the TPnA PWM duty and CCRB controls the TPnB PWM duty

5. CCRP will generate an interrupt request when the counter decrements to its zero value

6. n=3



ETM PWM Mode – Centre Aligned

Note: 1. Here TnCCLR=1 therefore CCRA clears the counter and determines the PWM period

2. TnPWM [1:0]=11 therefore the PWM is centre aligned

3. The internal PWM function continues running even when TnBIO [1:0]=00 or 01

4. CCRA controls the TPnB PWM period and CCRB controls the TPnB PWM duty

5. CCRP will generate an interrupt request when the counter decrements to its zero value

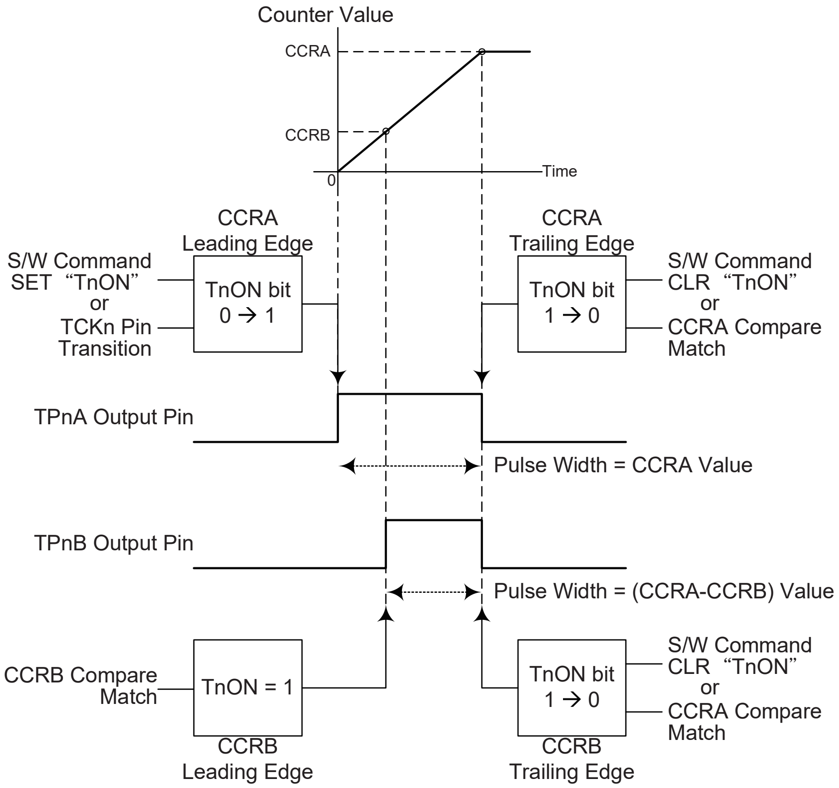
6. n=3

Single Pulse Output Mode

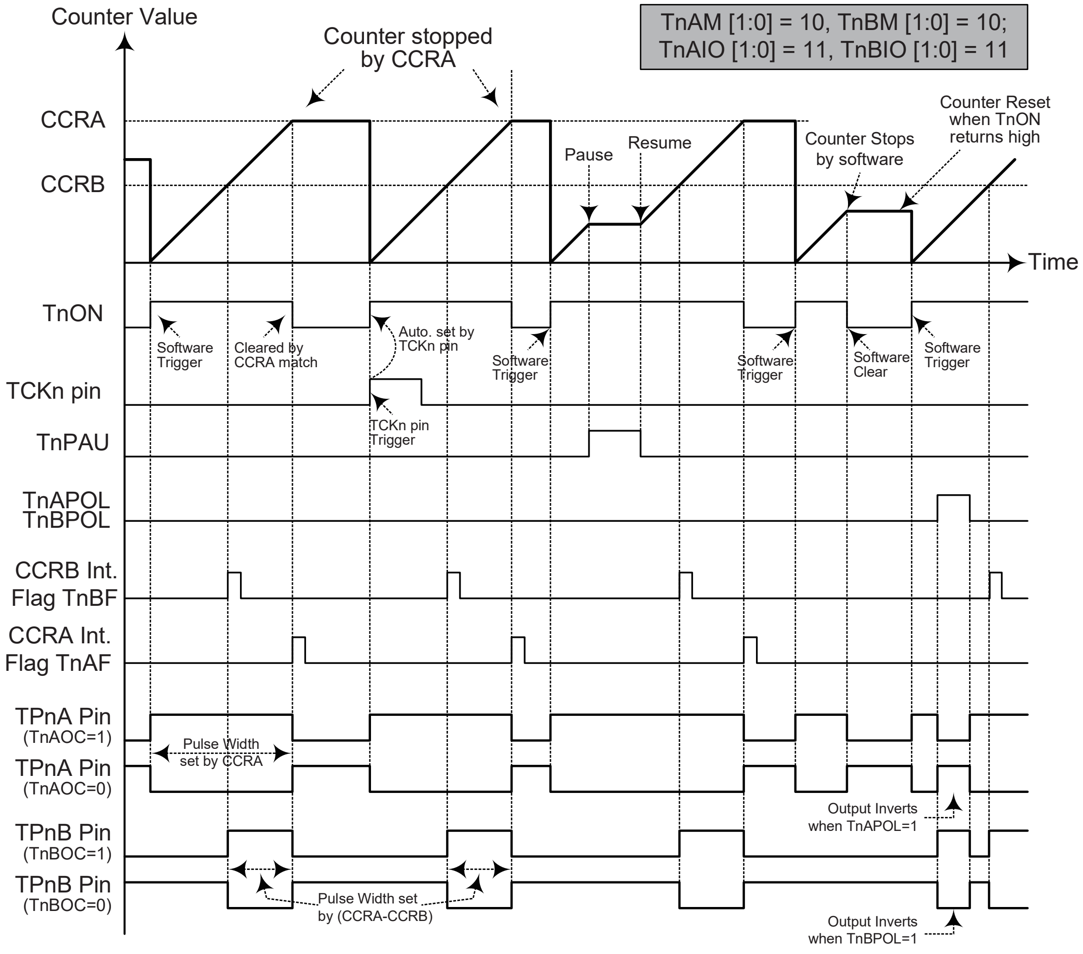
To select this mode, the required bit pairs, TnAM1, TnAM0 and TnBM1, TnBM0 should be set to 10 respectively and also the corresponding TnAIO1, TnAIO0 and TnBIO1, TnBIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the TM output pin.

The trigger for the pulse TPnA output leading edge is a low to high transition of the TnON bit, which can be implemented using the application program. The trigger for the pulse TPnB output leading edge is a compare match from Comparator B, which can be implemented using the application program. However in the Single Pulse Mode, the TnON bit can also be made to automatically change from low to high using the external TCKn pin, which will in turn initiate the Single Pulse output of TPnA. When the TnON bit transitions to a high level, the counter will start running and the pulse leading edge of TPnA will be generated. The TnON bit should remain high when the pulse is in its active state. The generated pulse trailing edge of TPnA and TPnB will be generated when the TnON bit is cleared to zero, which can be implemented using the application program or when a compare match occurs from Comparator A.

However a compare match from Comparator A will also automatically clear the TnON bit and thus generate the Single Pulse output trailing edge of TPnA and TPnB. In this way the CCRA value can be used to control the pulse width of TPnA. The CCRA-CCRB value can be used to control the pulse width of TPnB. A compare match from Comparator A and Comparator B will also generate TM interrupts. The counter can only be reset back to zero when the TnON bit changes from low to high when the counter restarts. In the Single Pulse Mode CCRP is not used. The TnCCLR bit is also not used.



Single Pulse Generation



Single Pulse Mode

Note: 1. Counter stopped by CCRA

2. CCRP is not used

3. The pulse is triggered by the TCKn pin or by setting the TnON bit high

4. A TCKn pin active edge will automatically set by the TnON bit high

5. In the Single Pulse Mode, TnAIO [1:0] and TnBIO [1:0] must be set to “11” and can not be changed

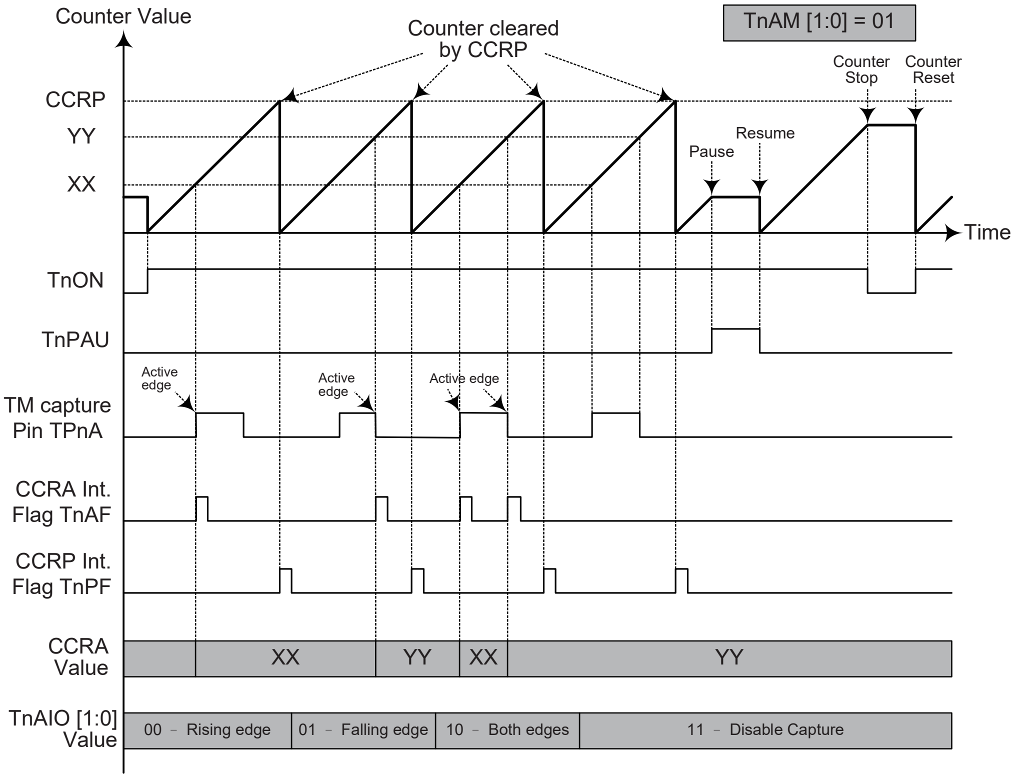
6. n=3

Capture Input Mode

To select this mode bits TnAM1, TnAM0 and TnBM1, TnBM0 in the TMnC1 and TMnC2 registers should be set to 01 respectively. This mode enables external signals to capture and store the present value of the internal counter and can therefore be used for applications such as pulse width measurements. The external signal is supplied on the TPnA and TPnB pins, whose active edge can be either a rising edge, a falling edge or both rising and falling edges; the active edge transition type is selected using the TnAIO1, TnAIO0 and TnBIO1, TnBIO0 bits in the TMnC1 and TMnC2 registers. The counter is started when the TnON bit changes from low to high which is initiated using the application program.

When the required edge transition appears on the TPnA and TPnB pins the present value in the counter will be latched into the CCRA and CCRB registers and a TM interrupt generated. Irrespective of what events occur on the TPnA and TPnB pins the counter will continue to free run until the TnON bit changes from high to low. When a CCRP compare match occurs the counter will reset back to zero; in this way the CCRP value can be used to control the maximum counter value. When a CCRP compare match occurs from Comparator P, a TM interrupt will also be generated. Counting the number of overflow interrupt signals from the CCRP can be a useful method in measuring long pulse widths. The TnAIO1, TnAIO0 and TnBIO1, TnBIO0 bits can select the active trigger edge on the TPnA and TPnB pins to be a rising edge, falling edge or both edge types. If the TnAIO1, TnAIO0 and TnBIO1, TnBIO0 bits are both set high, then no capture operation will take place irrespective of what happens on the TPnA and TPnB pins, however it must be noted that the counter will continue to run.

As the TPnA and TPnB pins are pin shared with other functions, care must be taken if the TM is in the Capture Input Mode. This is because if the pin is setup as an output, then any transitions on this pin may cause an input capture operation to be executed. The TnCCLR, TnAOC, TnBOC, TnAPOL and TnBPOL bits are not used in this mode.



ETM CCRA Capture Input Mode

Note: 1. TnAM [1:0]=01 and active edge set by the TnAIO [1:0] bits

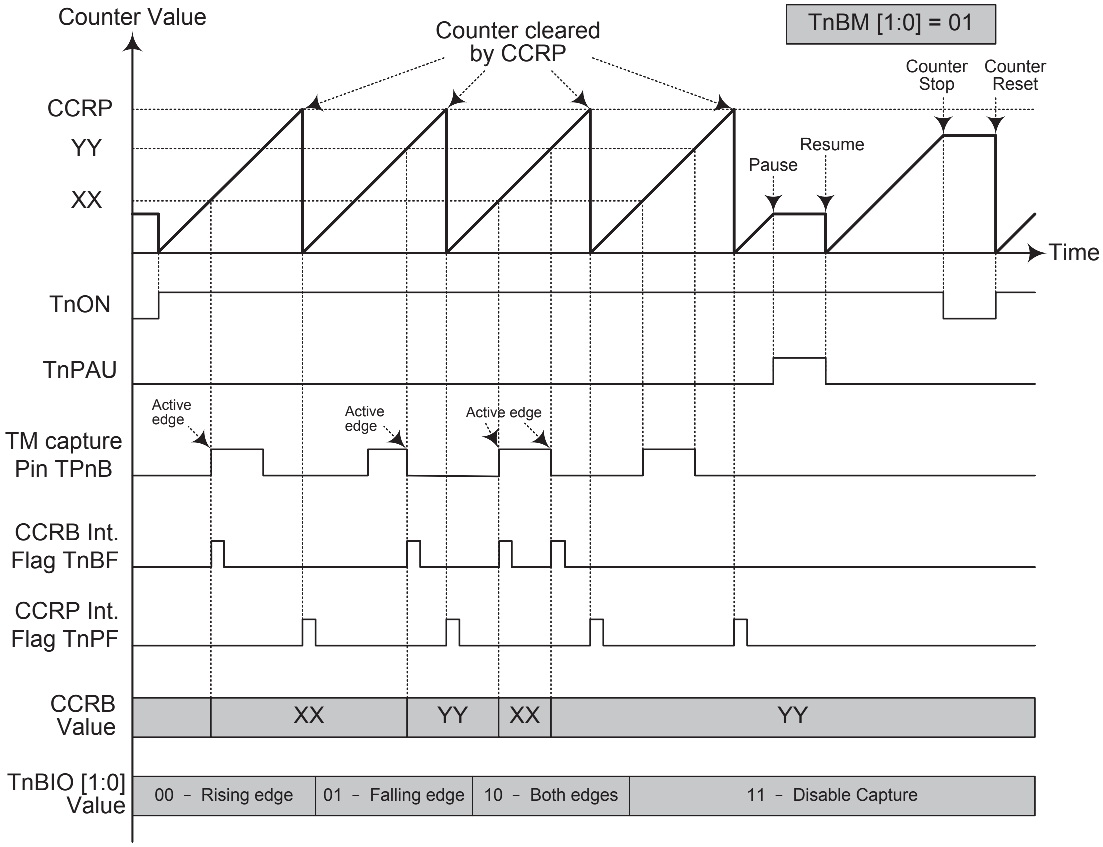
2. The TM Capture input pin active edge transfers the counter value to CCRA

3. TnCCLR bit not used

4. No output function – TnAOC and TnAPOL bits not used

5. CCRP determines the counter value and the counter has a maximum count value when CCRP is equal to zero

6. n=3



ETM CCRB Capture Input Mode

Note: 1. TnBM [1:0]=01 and active edge set by the TnBIO [1:0] bits

2. The TM Capture input pin active edge transfers the counter value to CCRB

3. TnCCLR bit not used

4. No output function – TnBOC and TnBPOL bits not used

5. CCRP determines the counter value and the counter has a maximum count value when CCRP is equal to zero

6. n=3

Analog to Digital Converter

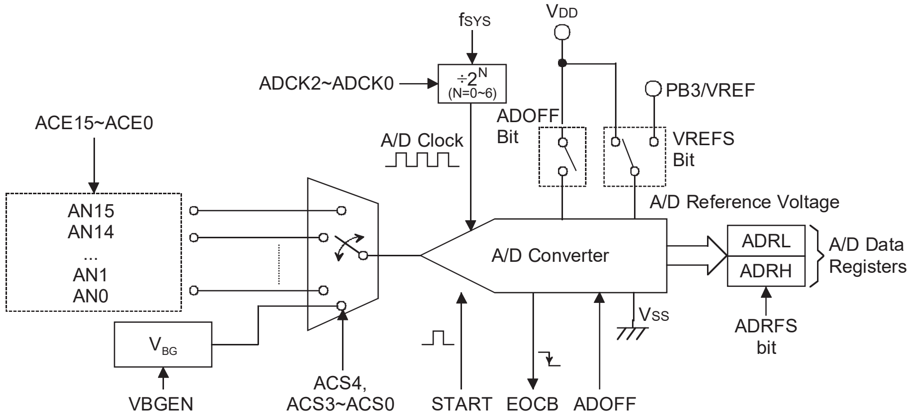
The need to interface to real world analog signals is a common requirement for many electronic systems. However, to properly process these signals by a microcontroller, they must first be converted into digital signals by A/D converters. By integrating the A/D conversion electronic circuitry into the microcontroller, the need for external components is reduced significantly with the corresponding follow-on benefits of lower costs and reduced component space requirements.

A/D Overview

The devices contain a multi-channel analog to digital converter which can directly interface to external analog signals, such as that from sensors or other control signals and convert these signals directly into either a 12-bit digital value.

Part No.	Input Channels	A/D Channel Select Bits	Input Pins
BC66F840	8	ACS4, ACS2~ACS0	AN0~AN7
BC66F850/ BC66F860	16	ACS4~ACS0	AN0~AN15

The accompanying block diagram shows the overall internal structure of the A/D converter, together with its associated registers.



A/D Converter Structure

A/D Converter Register Description

Overall operation of the A/D converter is controlled using five registers. A read only register pair exists to store the ADC data 12-bit value. The remaining three registers are control registers which setup the operating and control function of the A/D converter.

Register Name	Bit							
	7	6	5	4	3	2	1	0
ADRL (ADRFS=0)	D3	D2	D1	D0	—	—	—	—
ADRL (ADRFS=1)	D7	D6	D5	D4	D3	D2	D1	D0
ADRH (ADRFS=0)	D11	D10	D9	D8	D7	D6	D5	D4
ADRH (ADRFS=1)	—	—	—	—	D11	D10	D9	D8
ADCR0	START	EOCB	ADOFF	ADRFS	—	ACS2	ACS1	ACS0
ADCR1	ACS4	VBGEN	—	VREFS	—	ADCK2	ADCK1	ADCK0
ACERL	ACE7	ACE6	ACE5	ACE4	ACE3	ACE2	ACE1	ACE0

A/D Converter Register List – BC66F840

Register Name	Bit							
	7	6	5	4	3	2	1	0
ADRL (ADRFS=0)	D3	D2	D1	D0	—	—	—	—
ADRL (ADRFS=1)	D7	D6	D5	D4	D3	D2	D1	D0
ADRH (ADRFS=0)	D11	D10	D9	D8	D7	D6	D5	D4
ADRH (ADRFS=1)	—	—	—	—	D11	D10	D9	D8
ADCR0	START	EOCB	ADOFF	ADRFS	ACS3	ACS2	ACS1	ACS0
ADCR1	ACS4	VBGEN	—	VREFS	—	ADCK2	ADCK1	ADCK0
ACERL	ACE7	ACE6	ACE5	ACE4	ACE3	ACE2	ACE1	ACE0
ACERH	ACE15	ACE14	ACE13	ACE12	ACE11	ACE10	ACE9	ACE8

A/D Converter Register List – BC66F850/BC66F860

A/D Converter Data Registers – ADRL, ADRH

As the devices contain an internal 12-bit A/D converter, they require two data registers to store the converted value. These are a high byte register, known as ADRH, and a low byte register, known as ADRL. After the conversion process takes place, these registers can be directly read by the microcontroller to obtain the digitised conversion value. As only 12 bits of the 16-bit register space is utilised, the format in which the data is stored is controlled by the ADRFS bit in the ADCR0 register as shown in the accompanying table. D0~D11 are the A/D conversion result data bits. Any unused bits will be read as zero.

A/D Converter Control Registers – ADCR0, ADCR1, ACERL, ACERH

To control the function and operation of the A/D converter, up to four control registers known as ADCR0, ADCR1, ACERL and ACERH are provided. These 8-bit registers define functions such as the selection of which analog channel is connected to the internal A/D converter, the digitised data format, the A/D clock source as well as controlling the start function and monitoring the A/D converter end of conversion status. The ACS bit field in the ADCR0 register and ACS4 bit is the ADCR1 register define the ADC input channel number. As the devices contain only one actual analog to digital converter hardware circuit, each of the individual analog inputs must be routed to the converter. It is the function of the ACS4 and ACS bit field to determine which analog channel input pins or internal bandgap reference voltage is actually connected to the internal A/D converter.

The ACERL and ACERH control registers contain the ACE bit field which determines which pins on I/O ports are used as analog inputs for the A/D converter input and which pins are not to be used as the A/D converter input. Setting the corresponding bit high will select the A/D input function, clearing the bit to zero will select either the I/O or other pin-shared function. When the pin is selected to be an A/D input, its original function whether it is an I/O or other pin-shared function will be removed. In addition, any internal pull-high resistors connected to these pins will be automatically removed if the pin is selected to be an A/D input.

ADCR0 Register – BC66F840

Bit	7	6	5	4	3	2	1	0
Name	START	EOCB	ADOFF	ADRFS	—	ACS2	ACS1	ACS0
R/W	R/W	R	R/W	R/W	—	R/W	R/W	R/W
POR	0	1	1	0	—	0	0	0

Bit 7 **START:** Start the A/D conversion
0→1→0: start
0→1: Reset the A/D converter and set EOCB to “1”

This bit is used to initiate an A/D conversion process. The bit is normally low but if set high and then cleared low again, the A/D converter will initiate a conversion process. When the bit is set high the A/D converter will be reset.

Bit 6 **EOCB:** End of A/D conversion flag
0: A/D conversion ended
1: A/D conversion in progress

This read only flag is used to indicate when an A/D conversion process has completed. When the conversion process is running, the bit will be high.

Bit 5 **ADOFF:** A/D module on/off control bit
0: A/D module power on
1: A/D module power off

This bit controls the power to the A/D internal function. This bit should be cleared to zero to enable the A/D converter. If the bit is set high then the A/D converter will be switched off reducing the devices power consumption. As the A/D converter will consume a limited amount of power, even when not executing a conversion, this may be an important consideration in power sensitive battery powered applications.

Note: 1. It is recommended to set ADOFF=1 before entering IDLE/SLEEP Mode for saving power.
2. ADOFF=1 will power down the ADC module.

Bit 4 **ADRFS:** A/D Data Format control bit
0: ADC Data MSB is ADRH bit 7, LSB is ADRL bit 4
1: ADC Data MSB is ADRH bit 3, LSB is ADRL bit 0

This bit controls the format of the 12-bit converted A/D value in the two A/D data registers. Details are provided in the A/D data register section.

Bit 3 Unimplemented, read as “0”

Bit 2~0 **ACS2~ACS0:** Select A/D channel (when ACS4 is 0)
000: AN0
001: AN1
010: AN2
011: AN3
100: AN4
101: AN5
110: AN6
111: AN7

ADCR0 Register – BC66F850/BC66F860

Bit	7	6	5	4	3	2	1	0
Name	START	EOCB	ADOFF	ADRFS	ACS3	ACS2	ACS1	ACS0
R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	1	1	0	0	0	0	0

Bit 7 **START:** Start the A/D conversion
0→1→0: start
0→1: Reset the A/D converter and set EOCB to "1"

This bit is used to initiate an A/D conversion process. The bit is normally low but if set high and then cleared low again, the A/D converter will initiate a conversion process. When the bit is set high the A/D converter will be reset.

Bit 6 **EOCB:** End of A/D conversion flag
0: A/D conversion ended
1: A/D conversion in progress

This read only flag is used to indicate when an A/D conversion process has completed. When the conversion process is running, the bit will be high.

Bit 5 **ADOFF:** A/D module on/off control bit
0: A/D module power on
1: A/D module power off

This bit controls the power to the A/D internal function. This bit should be cleared to zero to enable the A/D converter. If the bit is set high then the A/D converter will be switched off reducing the devices power consumption. As the A/D converter will consume a limited amount of power, even when not executing a conversion, this may be an important consideration in power sensitive battery powered applications.

Note: 1. It is recommended to set ADOFF=1 before entering IDLE/SLEEP Mode for saving power.
2. ADOFF=1 will power down the ADC module.

Bit 4 **ADRFS:** A/D Data Format control bit
0: ADC Data MSB is ADRH bit 7, LSB is ADRL bit 4
1: ADC Data MSB is ADRH bit 3, LSB is ADRL bit 0

This bit controls the format of the 12-bit converted A/D value in the two A/D data registers. Details are provided in the A/D data register section.

Bit 3~0 **ACS3~ACS0:** Select A/D channel (when ACS4 is 0)
0000: AN0
0001: AN1
0010: AN2
0011: AN3
0100: AN4
0101: AN5
0110: AN6
0111: AN7
1000: AN8
1001: AN9
1010: AN10
1011: AN11
1100: AN12
1101: AN13
1110: AN14
1111: AN15

ADCR1 Register

Bit	7	6	5	4	3	2	1	0
Name	ACS4	VBGEN	—	VREFS	—	ADCK2	ADCK1	ADCK0
R/W	R/W	R/W	—	R/W	—	R/W	R/W	R/W
POR	0	0	—	0	—	0	0	0

Bit 7 **ACS4:** Select internal bandgap reference voltage as ADC input Control
0: Disable
1: Enable

This bit enables the internal bandgap reference voltage to be connected to the A/D converter. The VBGEN bit must first have been set to enable the bandgap circuit reference voltage to be used by the A/D converter. When the ACS4 bit is set high, the Bandgap reference voltage will be routed to the A/D converter and the other A/D input channels disconnected.

Bit 6 **VBGEN:** Internal Bandgap reference voltage Control
0: Disable
1: Enable

This bit controls the internal Bandgap circuit on/off function to the A/D converter. When the bit is set high the bandgap voltage can be used by the A/D converter. If the bandgap voltage is not used by the A/D converter and the LVR/LVD function is disabled then the bandgap reference circuit will be automatically switched off to conserve power. When the bandgap reference voltage is switched on for use by the A/D converter, a time t_{BG} should be allowed for the bandgap circuit to stabilise before implementing an A/D conversion.

Bit 5 Unimplemented, read as "0"

Bit 4 **VREFS:** Select ADC reference voltage
0: Internal ADC power
1: VREF pin

This bit is used to select the reference voltage for the A/D converter. If the bit is high, then the A/D converter reference voltage is supplied on the external VREF pin. If the pin is low, then the internal reference is used which is taken from the power supply pin VDD.

Bit 3 Unimplemented, read as "0"

Bit 2~0 **ADCK2~ADCK0:** Select ADC converter clock source
000: f_{SYS}
001: $f_{SYS}/2$
010: $f_{SYS}/4$
011: $f_{SYS}/8$
100: $f_{SYS}/16$
101: $f_{SYS}/32$
110: $f_{SYS}/64$
111: Undefined, can not be used

These three bits are used to select the clock source for the A/D converter.

ACERL Register

Bit	7	6	5	4	3	2	1	0
Name	ACE7	ACE6	ACE5	ACE4	ACE3	ACE2	ACE1	ACE0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

Bit 7 **ACE7:** Define AN7 is A/D input or not
0: Not A/D input
1: A/D input, AN7

Bit 6 **ACE6:** Define AN6 is A/D input or not
0: Not A/D input
1: A/D input, AN6

Bit 5 **ACE5:** Define AN5 is A/D input or not
0: Not A/D input
1: A/D input, AN5

Bit 4 **ACE4:** Define AN4 is A/D input or not
0: Not A/D input
1: A/D input, AN4

Bit 3 **ACE3:** Define AN3 is A/D input or not
0: Not A/D input
1: A/D input, AN3

Bit 2 **ACE2:** Define AN2 is A/D input or not
0: Not A/D input
1: A/D input, AN2

Bit 1 **ACE1:** Define AN1 is A/D input or not
0: Not A/D input
1: A/D input, AN1

Bit 0 **ACE0:** Define AN0 is A/D input or not
0: Not A/D input
1: A/D input, AN0

ACERH Register – BC66F850/BC66F860

Bit	7	6	5	4	3	2	1	0
Name	ACE15	ACE14	ACE13	ACE12	ACE11	ACE10	ACE9	ACE8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

Bit 7 **ACE15:** Define AN15 is A/D input or not
0: Not A/D input
1: A/D input, AN15

Bit 6 **ACE14:** Define AN14 is A/D input or not
0: Not A/D input
1: A/D input, AN14

Bit 5 **ACE13:** Define AN13 is A/D input or not
0: Not A/D input
1: A/D input, AN13

Bit 4 **ACE12:** Define AN12 is A/D input or not
0: Not A/D input
1: A/D input, AN12

Bit 3 **ACE11:** Define AN11 is A/D input or not
0: Not A/D input
1: A/D input, AN11

Bit 2 **ACE10:** Define AN10 is A/D input or not
0: Not A/D input
1: A/D input, AN10

Bit 1 **ACE9:** Define AN9 is A/D input or not
0: Not A/D input
1: A/D input, AN9

Bit 0 **ACE8:** Define AN8 is A/D input or not
0: Not A/D input
1: A/D input, AN8

A/D Operation

The START bit in the ADCR0 register is used to start and reset the A/D converter. When the microcontroller sets this bit from low to high and then low again, an analog to digital conversion cycle will be initiated. When the START bit is brought from low to high but not low again, the EOCB bit in the ADCR0 register will be set high and the analog to digital converter will be reset. It is the START bit that is used to control the overall start operation of the internal analog to digital converter.

The EOCB bit in the ADCR0 register is used to indicate when the analog to digital conversion process is complete. This bit will be automatically set to 0 by the microcontroller after a conversion cycle has ended. In addition, the corresponding A/D interrupt request flag will be set in the interrupt control register, and if the interrupts are enabled, an appropriate internal interrupt signal will be generated. This A/D internal interrupt signal will direct the program flow to the associated A/D internal interrupt address for processing. If the A/D internal interrupt is disabled, the microcontroller can be used to poll the EOCB bit in the ADCR0 register to check whether it has been cleared as an alternative method of detecting the end of an A/D conversion cycle.

The clock source for the A/D converter, which originates from the system clock f_{SYS}, can be chosen to be either f_{SYS} or a subdivided version of f_{SYS}. The division ratio value is determined by the ADCK2~ADCK0 bits in the ADCR1 register.

Although the A/D clock source is determined by the system clock f_{SYS}, and by bits ADCK2~ADCK0, there are some limitations on the A/D clock source speed range that can be selected. As the recommended range of permissible A/D clock period, t_{ADCK}, is from 0.5μs to 10μs, care must be taken for selected system clock frequencies. For example, if the system clock operates at a frequency of 4MHz, the ADCK2~ADCK0 bits should not be set to 000B or 110B. Doing so will give A/D clock periods that are less than the minimum A/D clock period or greater than the maximum A/D clock period which may result in inaccurate A/D conversion values. Refer to the following table for examples, where values marked with an asterisk * show where, depending upon the devices, special care must be taken, as the values may be less than the specified minimum A/D Clock Period.

f_{SYS}	A/D Clock Period (t_{ADCK})							
	ADCK2, ADCK1, ADCK0 =000 (f_{SYS})	ADCK2, ADCK1, ADCK0 =001 (f_{SYS}/2)	ADCK2, ADCK1, ADCK0 =010 (f_{SYS}/4)	ADCK2, ADCK1, ADCK0 =011 (f_{SYS}/8)	ADCK2, ADCK1, ADCK0 =100 (f_{SYS}/16)	ADCK2, ADCK1, ADCK0 =101 (f_{SYS}/32)	ADCK2, ADCK1, ADCK0 =110 (f_{SYS}/64)	ADCK2, ADCK1, ADCK0 =111
1MHz	1μs	2μs	4μs	8μs	16μs*	32μs*	64μs*	Undefined
2MHz	500ns	1μs	2μs	4μs	8μs	16μs*	32μs*	Undefined
4MHz	250ns*	500ns	1μs	2μs	4μs	8μs	16μs*	Undefined
8MHz	125ns*	250ns*	500ns	1μs	2μs	4μs	8μs	Undefined
12MHz	83ns*	167ns*	333ns*	667ns	1.33μs	2.67μs	5.33μs	Undefined
16MHz	63ns*	125ns*	250ns*	500ns	1μs	2μs	4μs	Undefined

A/D Clock Period Examples

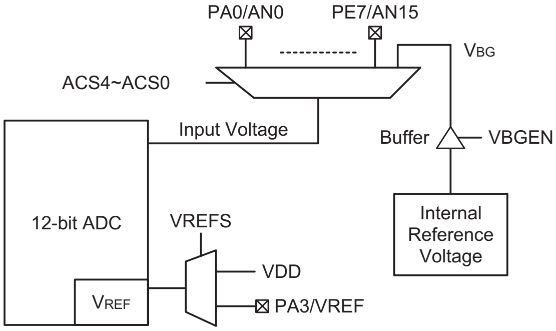
Controlling the power on/off function of the A/D converter circuitry is implemented using the ADOFF bit in the ADCR0 register. This bit must be zero to power on the A/D converter. When the ADOFF bit is cleared to zero to power on the A/D converter internal circuitry a certain delay, as indicated in the timing diagram, must be allowed before an A/D conversion is initiated. Even if no pins are selected for use as A/D inputs by clearing the ACE15~ACE0 bits in the ACERL and ACERH registers, if the ADOFF bit is zero then some power will still be consumed. In power conscious applications it is therefore recommended that the ADOFF is set high to reduce power consumption when the A/D converter function is not being used.

The reference voltage supply to the A/D Converter can be supplied from either the positive power supply pin, VDD, or from an external reference sources supplied on pin VREF. The desired selection is made using the VREFS bit. As the VREF pin is pin-shared with other functions, when the VREFS bit is set high, the VREF pin function will be selected and the other pin functions will be disabled automatically.

A/D Input Pins

All of the A/D analog input pins are pin-shared with the I/O pins as well as other functions. The ACE7~ACE0 bits in the ACERL register together with the ACE15~ACE8 bits in the ACERH register for BC66F850 and BC66F860 devices determine whether the input pins are setup as A/D converter analog inputs or whether they have other functions. If the ACE15~ACE0 bits for its corresponding pin is set high then the pin will be setup to be an A/D converter input and the original pin functions disabled. In this way, pins can be changed under program control to change their function between A/D inputs and other functions. All pull-high resistors, which are setup through register programming, will be automatically disconnected if the pins are setup as A/D inputs. Note that it is not necessary to first setup the A/D pin as an input in the port control register to enable the A/D input as when the ACE15~ACE0 bits enable an A/D input, the status of the port control register will be overridden.

The A/D converter has its own reference voltage pin VREF however the reference voltage can also be supplied from the power supply pin, a choice which is made through the VREFS bit in the ADCR1 register. The analog input values must not be allowed to exceed the value of V_{REF}.



A/D Input Structure

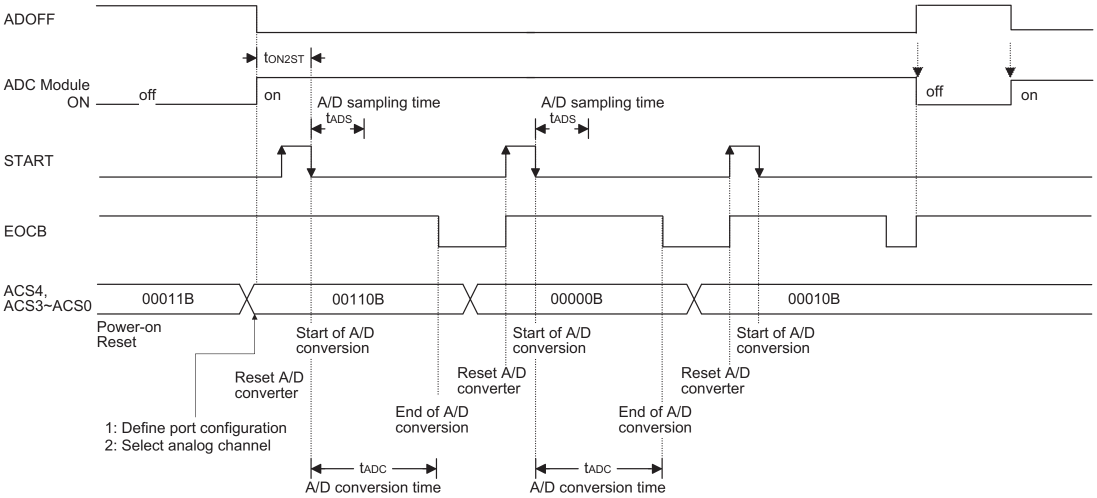
Summary of A/D Conversion Steps

The following summarises the individual steps that should be executed in order to implement an A/D conversion process.

- Step 1
  Select the required A/D conversion clock by correctly programming bits ADCK2~ADCK0 in the ADCR1 register.
- Step 2
  Enable the A/D by clearing the ADOFF bit in the ADCR0 register to zero.
- Step 3
  Select which channel is to be connected to the internal A/D converter by correctly programming the ACS4 and ACS3~ACS0 bits which are also contained in the ADCR1 and ADCR0 registers.
- Step 4
  Select which pins are to be used as A/D inputs and configure them by correctly programming the ACE15~ACE0 bits in the ACERL and ACERH registers.
- Step 5
  If the interrupts are to be used, the interrupt control registers must be correctly configured to ensure the A/D converter interrupt function is active. The master interrupt control bit, EMI, and the A/D converter interrupt bit, ADE, must both be set to high to do this.
- Step 6
  The analog to digital conversion process can now be initialised by setting the START bit in the ADCR0 register from low to high and then to low again. Note that this bit should have been originally cleared to 0.
- Step 7
  To check when the analog to digital conversion process is complete, the EOCB bit in the ADCR0 register can be polled. The conversion process is complete when this bit goes low. When this occurs, the A/D data registers ADRL and ADRH can be read to obtain the conversion value. As an alternative method if the interrupts are enabled and the stack is not full, the program can wait for an A/D interrupt to occur.

  Note: When checking for the end of the conversion process, if the method of polling the EOCB bit in the ADCR0 register is used, the interrupt enable step above can be omitted.

The accompanying diagram shows graphically the various stages involved in an analog to digital conversion process and its associated timing. After an A/D conversion process has been initiated by the application program, the microcontroller internal hardware will begin to carry out the conversion, during which time the program can continue with other functions. The time taken for the A/D conversion is 16 t_{ADCK} where t_{ADCK} is equal to the A/D clock period.



A/D Conversion Timing

Programming Considerations

During microcontroller operates where the A/D converter is not being used, the A/D internal circuitry can be switched off to reduce power consumption, by setting bit ADOFF high in the ADCR0 register. When this happens, the internal A/D converter circuits will not consume power irrespective of what analog voltage is applied to their input lines. If the A/D converter input lines are used as normal I/Os, then care must be taken as if the input voltage is not at a valid logic level, then this may lead to some increase in power consumption.

The power-on reset condition of the A/D converter control registers will ensure that the shared function pins are setup as A/D converter inputs. If any of the A/D converter input pins are to be used for other functions, then the A/D converter control register bits must be properly setup to disable the A/D input configuration.

A/D Transfer Function

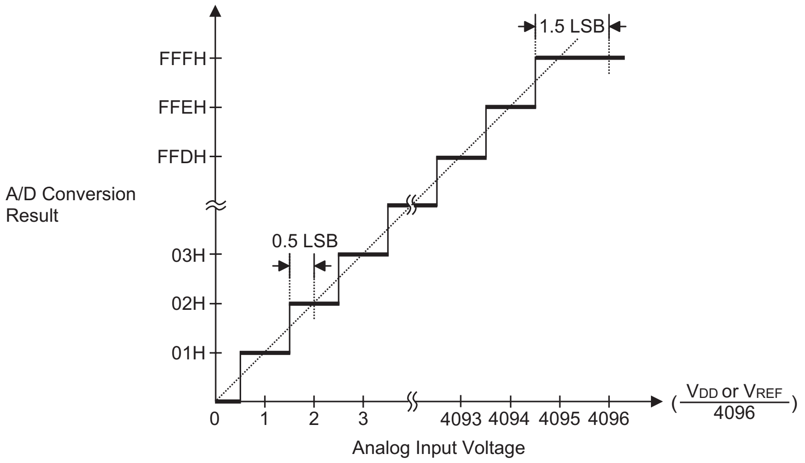
As the devices contain a 12-bit A/D converter, its full-scale converted digitised value is equal to FFFH. Since the full-scale analog input value is equal to the V_{DD} or V_{REF} voltage, this gives a single bit analog input value of V_{DD} or V_{REF} divided by 4096.

$$1\ \text{LSB} = (V_{DD}\ \text{or}\ V_{REF}) \div 4096$$

The A/D Converter input voltage value can be calculated using the following equation:

$$\text{A/D input voltage} = \text{A/D output digital value} \times (V_{DD}\ \text{or}\ V_{REF}) \div 4096$$

The diagram shows the ideal transfer function between the analog input value and the digitised output value for the A/D converter. Except for the digitised zero value, the subsequent digitised values will change at a point 0.5 LSB below where they would change without the offset, and the last full scale digitised value will change at a point 1.5 LSB below the V_{DD} or V_{REF} level.



Ideal A/D Transfer Function

A/D Programming Examples

The following two programming examples illustrate how to setup and implement an A/D conversion. In the first example, the method of polling the EOCB bit in the ADCR0 register is used to detect when the conversion cycle is complete, whereas in the second example, the A/D interrupt is used to determine when the conversion is complete.

Example 1: using an EOCB polling method to detect the end of conversion

```
clr ADE              ; disable ADC interrupt
mov a,03H
mov ADCR1,a          ; select fSYS/8 as A/D clock and switch off the internal Bandgap
                     ; reference voltage
clr ADOFF
mov a,0Fh            ; setup ACERL to configure pins AN0~AN3
mov ACERL,a
mov a,00h
mov ADCR0,a          ; enable and connect AN0 channel to A/D converter
:
:
start_conversion:
clr START            ; high pulse on start bit to initiate conversion
set START            ; reset A/D
clr START            ; start A/D

polling_EOC:
sz EOCB              ; poll the ADCR0 register EOCB bit to detect end of A/D conversion
jmp polling_EOC      ; continue polling
mov a,ADRL           ; read low byte conversion result value
mov ADRL_buffer,a    ; save result to user defined register
mov a,ADRH           ; read high byte conversion result value
mov ADRH_buffer,a    ; save result to user defined register
:
:
jmp start_conversion ; start next a/d conversion
```

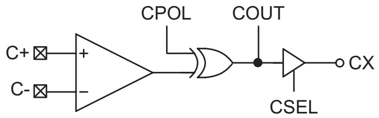
Example 2: using the interrupt method to detect the end of conversion

```
clr ADE                ; disable ADC interrupt
mov a,03H
mov ADCR1,a            ; select fSYS/8 as A/D clock and switch off the internal Bandgap
                       ; reference voltage
clr ADOFF
mov a,0Fh              ; setup ACERL to configure pins AN0~AN3
mov ACERL,a
mov a,00h
mov ADCR0,a            ; enable and connect AN0 channel to A/D converter

Start_conversion:
clr START              ; high pulse on START bit to initiate conversion
set START              ; reset A/D
clr START              ; start A/D
clr ADF                ; clear ADC interrupt request flag
set ADE                ; enable ADC interrupt
set EMI                ; enable global interrupt
:
:
                       ; ADC interrupt service routine
ADC_ISR:
mov acc_stack,a        ; save ACC to user defined memory
mov a,STATUS
mov status_stack,a     ; save STATUS to user defined memory
:
:
mov a,ADRL             ; read low byte conversion result value
mov adrl_buffer,a      ; save result to user defined register
mov a,ADRH             ; read high byte conversion result value
mov adrh_buffer,a      ; save result to user defined register
:
:
EXIT_INT_ISR:
mov a,status_stack
mov STATUS,a           ; restore STATUS from user defined memory
mov a,acc_stack        ; restore ACC from user defined memory
reti
```

Comparators

An analog comparator is contained within these devices. These functions offer flexibility via their register controlled features such as power-down, polarity select, hysteresis etc. In sharing their pins with normal I/O pins the comparators do not waste precious I/O pins if there functions are otherwise unused.



Comparator

Comparator Operation

The devices contain a comparator function which is used to compare two analog voltages and provide an output based on their difference. Full control over the internal comparators is provided via the control register CPC assigned to the comparator. The comparator output is recorded via a bit in the control register, but can also be transferred out onto a shared I/O pin. Additional comparator functions include, output polarity, hysteresis functions and power down control.

Any pull-high resistors connected to the shared comparator input pins will be automatically disconnected when the comparator is enabled. As the comparator inputs approach their switching level, some spurious output signals may be generated on the comparator output due to the slow rising or falling nature of the input signals. This can be minimised by selecting the hysteresis function will apply a small amount of positive feedback to the comparator. Ideally the comparator should switch at the point where the positive and negative inputs signals are at the same voltage level, however, unavoidable input offsets introduce some uncertainties here. The hysteresis function, if enabled, also increases the switching offset value.

Comparator Interrupt

The comparator possesses its own interrupt function. When the comparator output changes state, its relevant interrupt flag will be set, and if the corresponding interrupt enable bit is set, then a jump to its relevant interrupt vector will be executed. Note that it is the changing state of the COUT bit and not the output pin which generates an interrupt. If the microcontroller is in the SLEEP or IDLE Mode and the Comparator is enabled, then if the external input lines cause the Comparator output to change state, the resulting generated interrupt flag will also generate a wake-up. If it is required to disable a wake-up from occurring, then the interrupt flag should be first set high before entering the SLEEP or IDLE Mode.

Programming Considerations

If the comparator is enabled, it will remain active when the microcontroller enters the SLEEP or IDLE Mode, however as it will consume a certain amount of power, the user may wish to consider disabling it before the SLEEP or IDLE Mode is entered.

As comparator pins are shared with normal I/O pins the I/O registers for these pins will be read as zero (port control register is "1") or read as port data register value (port control register is "0") if the comparator function is enabled.

CPC Register

Bit	7	6	5	4	3	2	1	0
Name	CSEL	CEN	CPOL	COUT	COS	—	—	CHYEN
R/W	R/W	R/W	R/W	R/W	R/W	—	—	R/W
POR	1	0	0	0	0	—	—	1

Bit 7 **CSEL**: Select Comparator pins or I/O pins
0: I/O pin select
1: Comparator pin select

This is the Comparator pin or I/O pin select bit. If the bit is high the comparator will be selected and the two comparator input pins will be enabled. As a result, these two pins will lose their I/O pin functions. Any pull-high configuration options associated with the comparator shared pins will also be automatically disconnected.

Bit 6 **CEN**: Comparator On/Off control
0: Off
1: On

This is the Comparator on/off control bit. If the bit is zero the comparator will be switched off and no power consumed even if analog voltages are applied to its inputs. For power sensitive applications this bit should be cleared to zero if the comparator is not used or before the device enters the SLEEP or IDLE mode.

Bit 5 **CPOL**: Comparator output polarity
0: Output not inverted
1: Output inverted

This is the comparator polarity bit. If the bit is zero then the COUT bit will reflect the non-inverted output condition of the comparator. If the bit is high the comparator COUT bit will be inverted.

Bit 4 **COUT:** Comparator output bit
CPOL=0
0: C+ < C-
1: C+ > C-
CPOL=1
0: C+ > C-
1: C+ < C-

This bit stores the comparator output bit. The polarity of the bit is determined by the voltages on the comparator inputs and by the condition of the CPOL bit.

Bit 3 **COS**: Output path select
0: CX pin
1: Internal use

This is the comparator output path select control bit. If the bit is set to "0" and the CSEL bit is "1" the comparator output is connected to an external CX pin. If the bit is set to "1" or the CSEL bit is "0" the comparator output signal is only used internally by the device allowing the shared comparator output pin to retain its normal I/O operation.

Bit 2~1 Unimplemented, read as “0”

Bit 0 **CHYEN:** Hysteresis Control
0: Off
1: On

This is the hysteresis control bit and if set high will apply a limited amount of hysteresis to the comparator, as specified in the Comparator Electrical Characteristics table. The positive feedback induced by hysteresis reduces the effect of spurious switching near the comparator threshold.

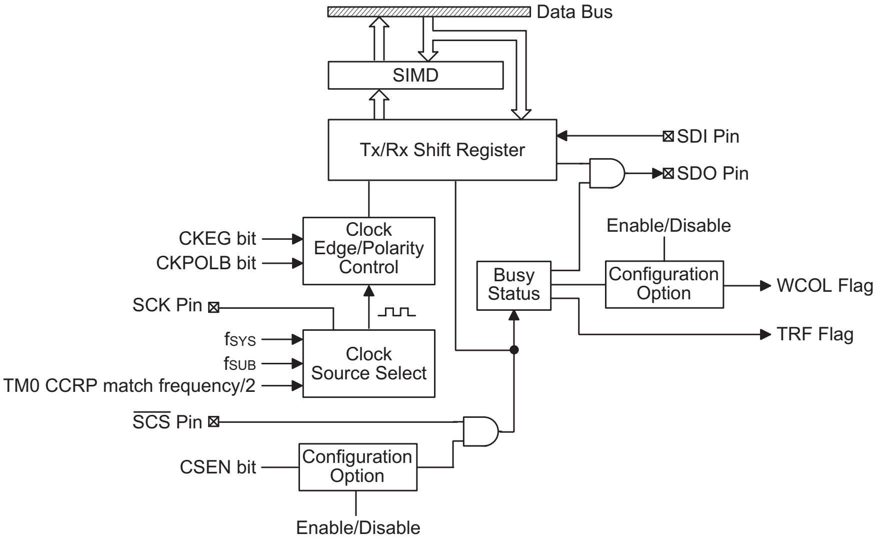
Serial Interface Module – SIM

These devices contain a Serial Interface Module, which includes both the four line SPI interface and the two line I^2C interface types, to allow an easy method of communication with external peripheral hardware. Having relatively simple communication protocols, these serial interface types allow the microcontroller to interface to external SPI or I^2C based hardware such as sensors, Flash or EEPROM memory, etc. The SIM interface pins are pin-shared with other I/O pins therefore the SIM interface function must first be selected using a configuration option. As both interface types share the same pins and registers, the choice of whether the SPI or I^2C type is used is made using the SIM operating mode control bits, named SIM2~SIM0, in the SIMC0 register. These pull-high resistors of the SIM pin-shared I/O pins are selected using pull-high control registers and also if the SIM function is enabled.

SPI Interface

The SPI interface is often used to communicate with external peripheral devices such as sensors, Flash or EEPROM memory devices etc. Originally developed by Motorola, the four line SPI interface is a synchronous serial data interface that has a relatively simple communication protocol simplifying the programming requirements when communicating with external hardware devices.

The communication is full duplex and operates as a slave/master type, where the devices can be either master or slave. Although the SPI interface specification can control multiple slave devices from a single master, these devices provided only one $\overline{SCS}$ pin. If the master needs to control multiple slave devices from a single master, the master can use I/O pin to select the slave devices.



SPI Block Diagram

SPI Interface Operation

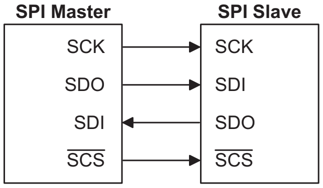
The SPI interface is a full duplex synchronous serial data link. It is a four line interface with pin names SDI, SDO, SCK and $\overline{\text{SCS}}$. Pins SDI and SDO are the Serial Data Input and Serial Data Output lines, SCK is the Serial Clock line and $\overline{\text{SCS}}$ is the Slave Select line. As the SPI interface pins are pin-shared with normal I/O pins and with the I^{2}C function pins, the SPI interface must first be enabled by selecting the SIM enable configuration option and setting the correct bits in the SIMC0 and SIMC2 registers. After the SPI configuration option has been configured it can also be additionally disabled or enabled using the SIMEN bit in the SIMC0 register. Communication between devices connected to the SPI interface is carried out in a slave/master mode with all data transfer initiations being implemented by the master. The Master also controls the clock signal. As the devices only contain a single $\overline{\text{SCS}}$ pin only one slave device can be utilized. The $\overline{\text{SCS}}$ pin is controlled by software, set CSEN bit to 1 to enable $\overline{\text{SCS}}$ pin function, set CSEN bit to 0 the $\overline{\text{SCS}}$ pin will be floating state.

The SPI function in these devices offer the following features:

- Full duplex synchronous data transfer
- Both Master and Slave modes
- LSB first or MSB first data transmission modes
- Transmission complete flag
- Rising or falling active clock edge
- WCOL and CSEN bit enabled or disable select

The status of the SPI interface pins is determined by a number of factors such as whether the devices are in the master or slave mode and upon the condition of certain control bits such as CSEN and SIMEN.

There are several configuration options associated with the SPI interface. One of these is to enable the SIM function which selects the SIM pins rather than normal I/O pins. Note that if the configuration option does not select the SIM function then the SIMEN bit in the SIMC0 register will have no effect. Another two SPI configuration options determine if the CSEN and WCOL bits are to be used.



SPI Master/Slave Connection

SPI Registers

There are three internal registers which control the overall operation of the SPI interface. These are the SIMD data register and two registers SIMC0 and SIMC2. Note that the SIMC1 register is only used by the I²C interface.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	PCKEN	PCKP1	PCKP0	SIMEN	—
SIMD	D7	D6	D5	D4	D3	D2	D1	D0
SIMC2	D7	D6	CKPOLB	CKEG	MLS	CSEN	WCOL	TRF

SIM Registers List

SIMD Register

The SIMD register is used to store the data being transmitted and received. The same register is used by both the SPI and I²C functions. Before the devices write data to the SPI bus, the actual data to be transmitted must be placed in the SIMD register. After the data is received from the SPI bus, the devices can read it from the SIMD register. Any transmission or reception of data from the SPI bus must be made via the SIMD register.

- **SIMD Register**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x": unknown

There are also two control registers for the SPI interface, SIMC0 and SIMC2. Note that the SIMC2 register also has the name SIMA which is used by the I²C function. The SIMC1 register is not used by the SPI function, only by the I²C function. Register SIMC0 is used to control the enable/disable function and to set the data transmission clock frequency. Although not connected with the SPI function, the SIMC0 register is also used to control the Peripheral Clock Prescaler. Register SIMC2 is used for other control functions such as LSB/MSB selection, write collision flag etc.

SIMC0 Register

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	PCKEN	PCKP1	PCKP0	SIMEN	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	—
POR	1	1	1	0	0	0	0	—

Bit 7~5 **SIM2, SIM1, SIM0:** SIM Operating Mode Control
000: SPI master mode; SPI clock is $f_{SYS}/4$
001: SPI master mode; SPI clock is $f_{SYS}/16$
010: SPI master mode; SPI clock is $f_{SYS}/64$
011: SPI master mode; SPI clock is f_{SUB}
100: SPI master mode; SPI clock is TM0 CCRP match frequency/2
101: SPI slave mode
110: I^2C slave mode
111: Unused mode

These bits setup the overall operating mode of the SIM function. As well as selecting if the I^2C or SPI function, they are used to control the SPI Master/Slave selection and the SPI Master clock frequency. The SPI clock is a function of the system clock but can also be chosen to be sourced from the TM0. If the SPI Slave Mode is selected then the clock will be supplied by an external Master device.

Bit 4 **PCKEN:** PCK Output Pin Control
0: Disable
1: Enable

Bit 3~2 **PCKP1, PCKP0**: Select PCK output pin frenquency
00: f_{SYS}
01: $f_{SYS}/4$
10: $f_{SYS}/8$
11: TM0 CCRP match frequency/2

Bit 1 **SIMEN:** SIM Control
0: Disable
1: Enable

The bit is the overall on/off control for the SIM interface. When the SIMEN bit is cleared to zero to disable the SIM interface, the SDI, SDO, SCK and $\overline{SCS}$, or SDA and SCL lines will be in a floating condition and the SIM operating current will be reduced to a minimum value. When the bit is high the SIM interface is enabled. The SIM configuration option must have first enabled the SIM interface for this bit to be effective. If the SIM is configured to operate as an SPI interface via the SIM2~SIM0 bits, the contents of the SPI control registers will remain at the previous settings when the SIMEN bit changes from low to high and should therefore be first initialised by the application program. If the SIM is configured to operate as an I^2C interface via the SIM2~SIM0 bits and the SIMEN bit changes from low to high, the contents of the I^2C control bits such as HTX and TXAK will remain at the previous settings and should therefore be first initialised by the application program while the relevant I^2C flags such as HCF, HAAS, HBB, SRW and RXAK will be set to their default states.

Bit 0 Unimplemented, read as "0"

SIMC2 Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	CKPOLB	CKEG	MLS	CSEN	WCOL	TRF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 Undefined bit
This bit can be read or written by the user software program.

Bit 5 **CKPOLB:** Determines the base condition of the clock line
0: The SCK line will be high when the clock is inactive
1: The SCK line will be low when the clock is inactive

The CKPOLB bit determines the base condition of the clock line, if the bit is high, then the SCK line will be low when the clock is inactive. When the CKPOLB bit is low, then the SCK line will be high when the clock is inactive.

Bit 4 **CKEG:** Determines SPI SCK active clock edge type
CKPOLB=0
0: SCK is high base level and data capture at SCK rising edge
1: SCK is high base level and data capture at SCK falling edge
CKPOLB=1
0: SCK is low base level and data capture at SCK falling edge
1: SCK is low base level and data capture at SCK rising edge

The CKEG and CKPOLB bits are used to setup the way that the clock signal outputs and inputs data on the SPI bus. These two bits must be configured before data transfer is executed otherwise an erroneous clock edge may be generated. The CKPOLB bit determines the base condition of the clock line, if the bit is high, then the SCK line will be low when the clock is inactive. When the CKPOLB bit is low, then the SCK line will be high when the clock is inactive. The CKEG bit determines active clock edge type which depends upon the condition of CKPOLB bit.

Bit 3 **MLS:** SPI Data shift order
0: LSB
1: MSB

This is the data shift select bit and is used to select how the data is transferred, either MSB or LSB first. Setting the bit high will select MSB first and low for LSB first.

Bit 2 **CSEN:** SPI $\overline{\text{SCS}}$ pin Control
0: Disable
1: Enable

The CSEN bit is used as an enable/disable for the $\overline{\text{SCS}}$ pin. If this bit is low, then the $\overline{\text{SCS}}$ pin will be disabled and placed into a floating condition. If the bit is high the $\overline{\text{SCS}}$ pin will be enabled and used as a select pin.

Note that using the CSEN bit can be disabled or enabled via configuration option.

Bit 1 **WCOL:** SPI Write Collision flag
0: No collision
1: Collision

The WCOL flag is used to detect if a data collision has occurred. If this bit is high it means that data has been attempted to be written to the SIMD register during a data transfer operation. This writing operation will be ignored if data is being transferred. The bit can be cleared by the application program.

Note that using the WCOL bit can be disabled or enabled via configuration option.

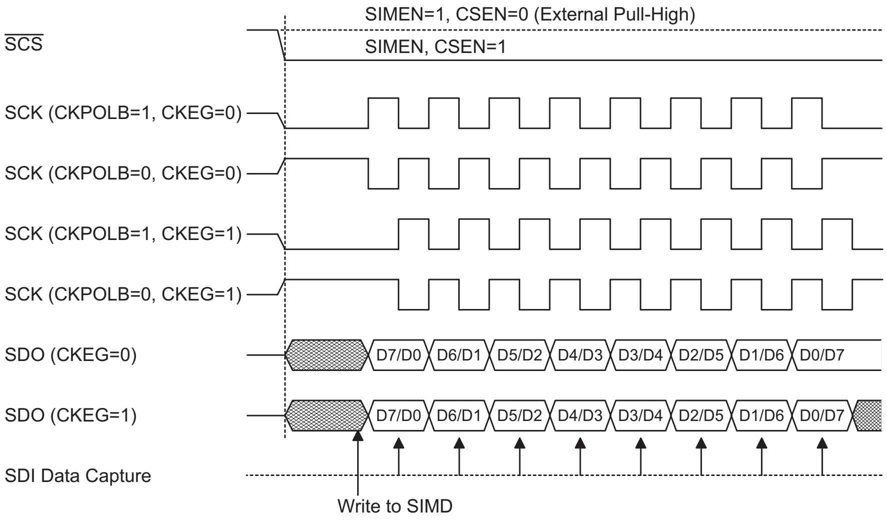
Bit 0 **TRF:** SPI Transmit/Receive Complete flag
0: Data is being transferred
1: SPI data transmission is completed

The TRF bit is the Transmit/Receive Complete flag and is set “1” automatically when an SPI data transmission is completed, but must set to “0” by the application program. It can be used to generate an interrupt.

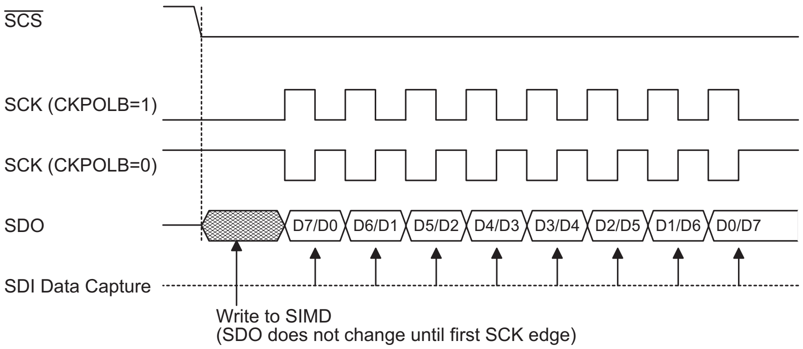
SPI Communication

After the SPI interface is enabled by setting the SIMEN bit high, then in the Master Mode, when data is written to the SIMD register, transmission/reception will begin simultaneously. When the data transfer is complete, the TRF flag will be set automatically, but must be cleared using the application program. In the Slave Mode, when the clock signal from the master has been received, any data in the SIMD register will be transmitted and any data on the SDI pin will be shifted into the SIMD register.

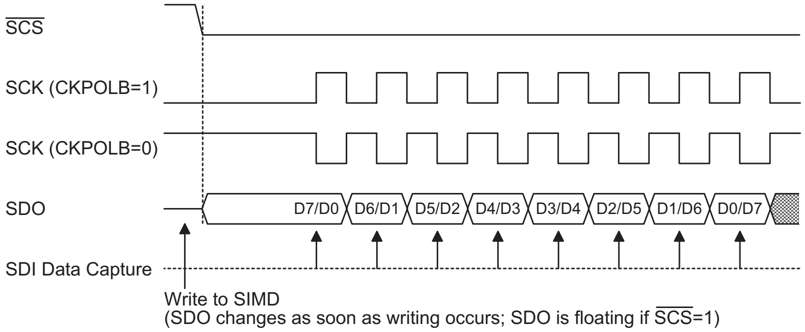
The master should output an $\overline{SCS}$ signal to enable the slave device before a clock signal is provided. The slave data to be transferred should be well prepared at the appropriate moment relative to the $\overline{SCS}$ signal depending upon the configurations of the CKPOLB bit and CKEG bit. The accompanying timing diagram shows the relationship between the slave data and $\overline{SCS}$ signal for various configurations of the CKPOLB and CKEG bits. The SPI will continue to function even in the IDLE Mode.



SPI Master Mode Timing

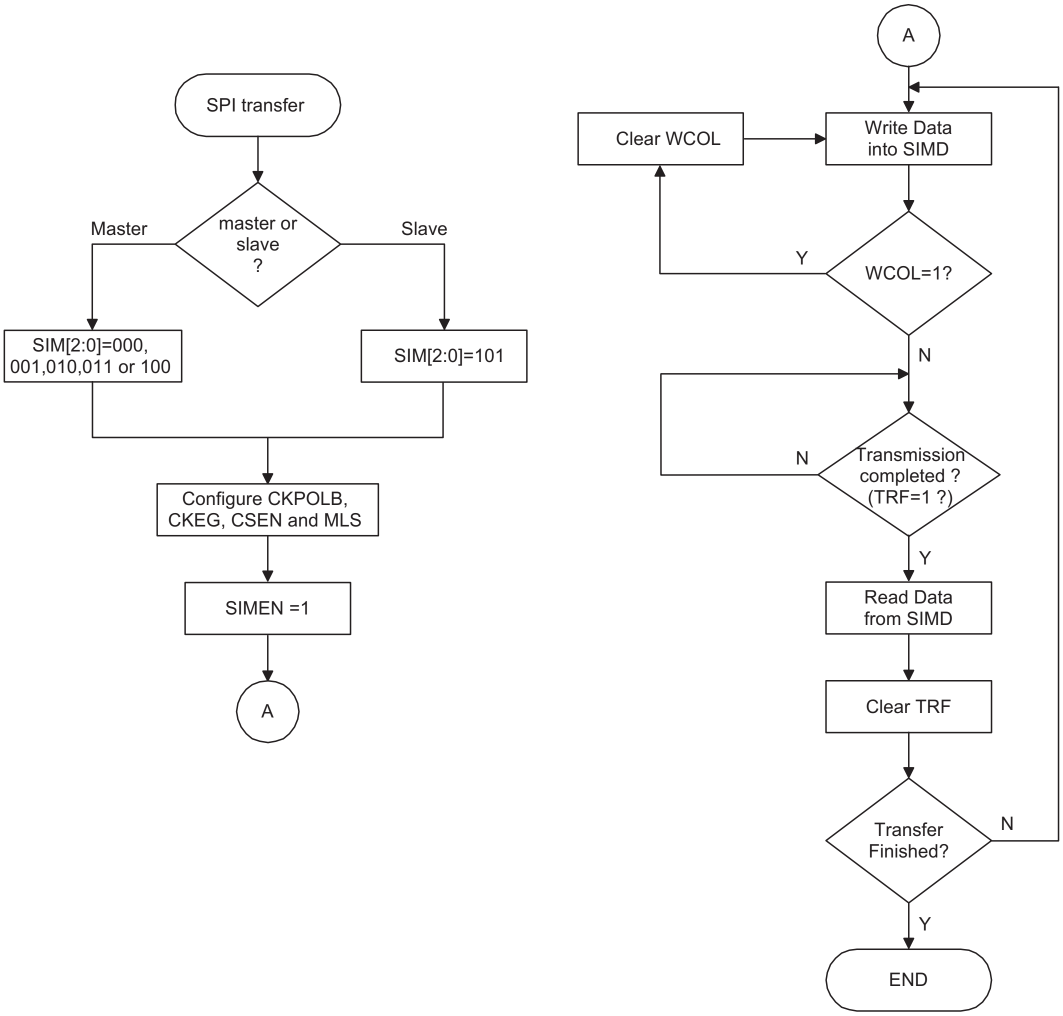


SPI Slave Mode Timing – CKEG=0



Note: For SPI slave mode, if SIMEN=1 and CSEN=0, SPI is always enabled and ignores the SCS level.

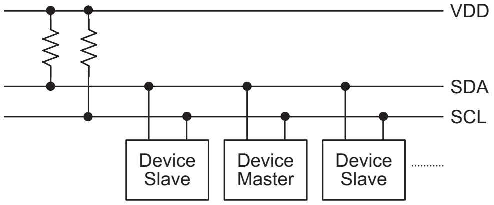
SPI Slave Mode Timing – CKEG=1



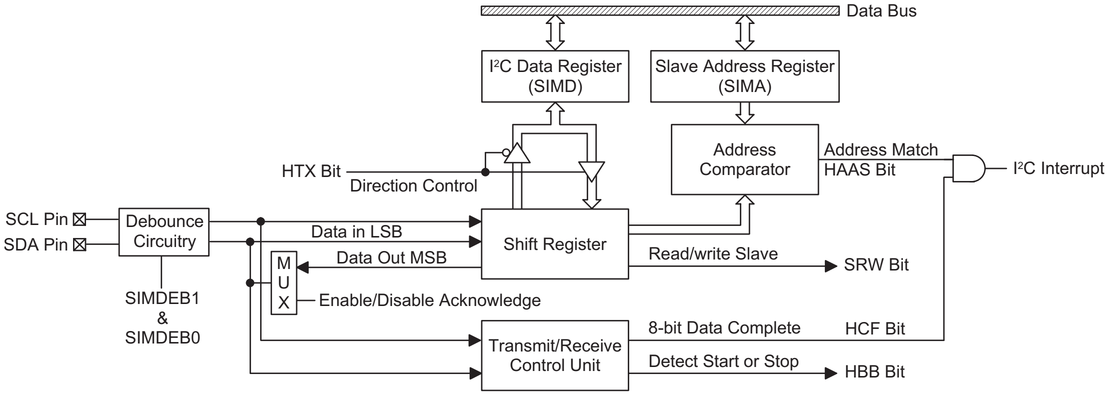
SPI Transfer Control Flowchart

I^{2}C Interface

The I^{2}C interface is used to communicate with external peripheral devices such as sensors, EEPROM memory etc. Originally developed by Philips, it is a two line low speed serial interface for synchronous serial data transfer. The advantage of only two lines for communication, relatively simple communication protocol and the ability to accommodate multiple devices on the same bus has made it an extremely popular interface type for many applications.



I^{2}C Master Slave Bus Connection



I^{2}C Block Diagram

I^{2}C Interface Operation

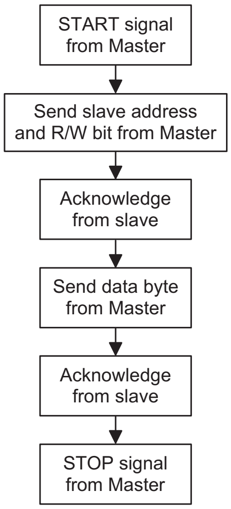
The I^{2}C serial interface is a two line interface, a serial data line, SDA, and serial clock line, SCL. As many devices may be connected together on the same bus, their outputs are both open drain types. For this reason it is necessary that external pull-high resistors are connected to these outputs. Note that no chip select line exists, as each device on the I^{2}C bus is identified by a unique address which will be transmitted and received on the I^{2}C bus.

When two devices communicate with each other on the bidirectional I^{2}C bus, one is known as the master device and one as the slave device. Both master and slave can transmit and receive data. However, it is the master device that has overall control of the bus. For these devices, which only operates in slave mode, there are two methods of transferring data on the I^{2}C bus, the slave transmit mode and the slave receive mode.

There are several configuration options associated with the I^{2}C interface. One of these is to enable the function which selects the SIM pins rather than normal I/O pins. Note that if the configuration option does not select the SIM function then the SIMEN bit in the SIMC0 register will have no effect. A configuration option determines the debounce time of the I^{2}C interface. This uses the system clock to in effect add a debounce time to the external clock to reduce the possibility of glitches on the clock line causing erroneous operation. The debounce time, if selected, can be chosen to be either 2 or 4 system clocks. To achieve the required I^{2}C data transfer speed, there exists a relationship between the system clock, f_{SYS}, and the I^{2}C debounce time. For either the I^{2}C Standard or Fast mode operation, users must take care of the selected system clock frequency and the configured debounce time to match the criterion shown in the following table.

I^{2}C Debounce Time Selection	I^{2}C Standard Mode (100kHz)	I^{2}C Fast Mode (400kHz)
No debounce	f_{SYS} > 2MHz	f_{SYS} > 5MHz
2 system clock debounce	f_{SYS} > 4MHz	f_{SYS} > 10MHz
4 system clock debounce	f_{SYS} > 8MHz	f_{SYS} > 20MHz

I^{2}C Minimum f_{SYS} Frequency



I^{2}C Registers

There are three control registers associated with the I^{2}C bus, SIMC0, SIMC1 and SIMA and one data register, SIMD. The SIMD register, which is shown in the above SPI section, is used to store the data being transmitted and received on the I^{2}C bus. Before the microcontroller writes data to the I^{2}C bus, the actual data to be transmitted must be placed in the SIMD register. After the data is received from the I^{2}C bus, the microcontroller can read it from the SIMD register. Any transmission or reception of data from the I^{2}C bus must be made via the SIMD register.

Note that the SIMA register also has the name SIMC2 which is used by the SPI function. Bit SIMEN and bits SIM2~SIM0 in register SIMC0 are used by the I^{2}C interface.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	PCKEN	PCKP1	PCKP0	SIMEN	—
SIMC1	HCF	HAAS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
SIMD	D7	D6	D5	D4	D3	D2	D1	D0
SIMA	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	—
I2CTOC	I2CTOEN	I2CTOF	I2CTOS5	I2CTOS4	I2CTOS3	I2CTOS2	I2CTOS1	I2CTOS0

I^{2}C Registers List

SIMC0 Register

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	PCKEN	PCKP1	PCKP0	SIMEN	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	—
POR	1	1	1	0	0	0	0	—

Bit 7~5 **SIM2, SIM1, SIM0:** SIM Operating Mode Control
000: SPI master mode; SPI clock is $f_{SYS}/4$
001: SPI master mode; SPI clock is $f_{SYS}/16$
010: SPI master mode; SPI clock is $f_{SYS}/64$
011: SPI master mode; SPI clock is f_{SUB}
100: SPI master mode; SPI clock is TM0 CCRP match frequency/2
101: SPI slave mode
110: I^{2}C slave mode
111: Unused mode

These bits setup the overall operating mode of the SIM function. As well as selecting if the I^{2}C or SPI function, they are used to control the SPI Master/Slave selection and the SPI Master clock frequency. The SPI clock is a function of the system clock but can also be chosen to be sourced from the TM0. If the SPI Slave Mode is selected then the clock will be supplied by an external Master device.

Bit 4 **PCKEN:** PCK Output Pin Control
0: Disable
1: Enable

Bit 3~2 **PCKP1, PCKP0**: Select PCK output pin frenquency
00: f_{SYS}
01: $f_{SYS}/4$
10: $f_{SYS}/8$
11: TM0 CCRP match frequency/2

Bit 1 **SIMEN:** SIM Control
0: Disable
1: Enable

The bit is the overall on/off control for the SIM interface. When the SIMEN bit is cleared to zero to disable the SIM interface, the SDI, SDO, SCK and $\overline{SCS}$, or SDA and SCL lines will be in a floating condition and the SIM operating current will be reduced to a minimum value. When the bit is high the SIM interface is enabled. The SIM configuration option must have first enabled the SIM interface for this bit to be effective. If the SIM is configured to operate as an SPI interface via the SIM2~SIM0 bits, the contents of the SPI control registers will remain at the previous settings when the SIMEN bit changes from low to high and should therefore be first initialised by the application program. If the SIM is configured to operate as an I^{2}C interface via the SIM2~SIM0 bits and the SIMEN bit changes from low to high, the contents of the I^{2}C control bits such as HTX and TXAK will remain at the previous settings and should therefore be first initialised by the application program while the relevant I^{2}C flags such as HCF, HAAS, HBB, SRW and RXAK will be set to their default states.

Bit 0 Unimplemented, read as “0”

SIMC1 Register

Bit	7	6	5	4	3	2	1	0
Name	HCF	HAAS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
R/W	R	R	R	R/W	R/W	R	R/W	R
POR	1	0	0	0	0	0	0	1

Bit 7 **HCF:** I^{2}C Bus data transfer completion flag
0: Data is being transferred
1: Completion of an 8-bit data transfer

The HCF flag is the data transfer flag. This flag will be zero when data is being transferred. Upon completion of an 8-bit data transfer the flag will go high and an interrupt will be generated.

Bit 6 **HAAS:** I^{2}C Bus address match flag
0: Not address match
1: Address match

The HAAS flag is the address match flag. This flag is used to determine if the slave device address is the same as the master transmit address. If the addresses match then this bit will be high, if there is no match then the flag will be low.

Bit 5 **HBB:** I^{2}C Bus busy flag
0: I^{2}C Bus is not busy
1: I^{2}C Bus is busy

The HBB flag is the I^{2}C busy flag. This flag will be “1” when the I^{2}C bus is busy which will occur when a START signal is detected. The flag will be set to “0” when the bus is free which will occur when a STOP signal is detected.

Bit 4 **HTX:** Select I^{2}C slave device is transmitter or receiver
0: Slave device is the receiver
1: Slave device is the transmitter

Bit 3 **TXAK:** I^{2}C Bus transmit acknowledge flag
0: Slave send acknowledge flag
1: Slave do not send acknowledge flag

The TXAK bit is the transmit acknowledge flag. After the slave device receipt of 8-bits of data, this bit will be transmitted to the bus on the 9th clock from the slave device. The slave device must always set TXAK bit to “0” before further data is received.

Bit 2 **SRW:** I^{2}C Slave Read/Write flag
0: Slave device should be in receive mode
1: Slave device should be in transmit mode

The SRW flag is the I^{2}C Slave Read/Write flag. This flag determines whether the master device wishes to transmit or receive data from the I^{2}C bus. When the transmitted address and slave address is match, that is when the HAAS flag is set high, the slave device will check the SRW flag to determine whether it should be in transmit mode or receive mode. If the SRW flag is high, the master is requesting to read data from the bus, so the slave device should be in transmit mode. When the SRW flag is zero, the master will write data to the bus, therefore the slave device should be in receive mode to read this data.

Bit 1 **IAMWU:** I^{2}C Address Match Wake-up Control
0: Disable
1: Enable – must be cleared by the application program after wake-up

This bit should be set to 1 to enable the I^{2}C address match wake up from the SLEEP or IDLE Mode. If the IAMWU bit has been set before entering either the SLEEP or IDLE mode to enable the I^{2}C address match wake up, then this bit must be cleared by the application program after wake-up to ensure correction device operation.

Bit 0 **RXAK:** I²C Bus Receive acknowledge flag
0: Slave receive acknowledge flag
1: Slave does not receive acknowledge flag

The RXAK flag is the receiver acknowledge flag. When the RXAK flag is "0", it means that a acknowledge signal has been received at the 9th clock, after 8 bits of data have been transmitted. When the slave device in the transmit mode, the slave device checks the RXAK flag to determine if the master receiver wishes to receive the next byte. The slave transmitter will therefore continue sending out data until the RXAK flag is "1". When this occurs, the slave transmitter will release the SDA line to allow the master to send a STOP signal to release the I²C Bus.

The SIMD register is used to store the data being transmitted and received. The same register is used by both the SPI and I²C functions. Before the devices write data to the SPI bus, the actual data to be transmitted must be placed in the SIMD register. After the data is received from the SPI bus, the devices can read it from the SIMD register. Any transmission or reception of data from the SPI bus must be made via the SIMD register.

SIMD Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	X	x	x	x	x	x	x

"x": unknown

SIMA Register

Bit	7	6	5	4	3	2	1	0
Name	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	—
POR	x	X	x	x	x	x	x	—

"x": unknown

Bit 7~1 **IICA6~IICA0:** I²C slave address
IICA6~IICA0 is the I²C slave address bit 6~bit 0

The SIMA register is also used by the SPI interface but has the name SIMC2. The SIMA register is the location where the 7-bit slave address of the slave device is stored. Bits 7~1 of the SIMA register define the device slave address. Bit 0 is not defined.

When a master device, which is connected to the I²C bus, sends out an address, which matches the slave address in the SIMA register, the slave device will be selected. Note that the SIMA register is the same register address as SIMC2 which is used by the SPI interface.

Bit 0 Unimplemented, read as "0"
This bit can be read or written by user software program.

I²C Bus Communication

Communication on the I²C bus requires four separate steps, a START signal, a slave device address transmission, a data transmission and finally a STOP signal. When a START signal is placed on the I²C bus, all devices on the bus will receive this signal and be notified of the imminent arrival of data on the bus. The first seven bits of the data will be the slave address with the first bit being the MSB. If the address of the slave device matches that of the transmitted address, the HAAS bit in the SIMC1 register will be set and an I²C interrupt will be generated. After entering the interrupt service routine, the slave device must first check the condition of the HAAS bit to determine whether the interrupt source originates from an address match or from the completion of an 8-bit data transfer. During a data transfer, note that after the 7-bit slave address has been transmitted, the following bit, which is the 8th bit, is the read/write bit whose value will be placed in the SRW bit. This bit will be checked by the slave device to determine whether to go into transmit or receive mode. Before any transfer of data to or from the I²C bus, the microcontroller must initialise the bus, the following are steps to achieve this:

Step 1

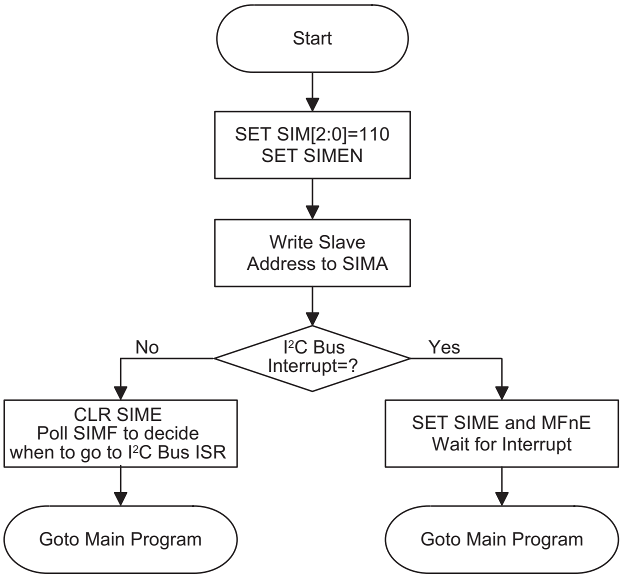
Set the SIM2~SIM0 and SIMEN bits in the SIMC0 register to 1 to enable the I²C bus.

Step 2

Write the slave address of the devices to the I²C bus address register SIMA.

Step 3

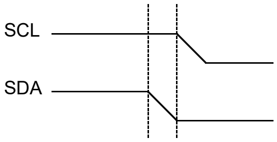
Set the SIME and SIM Multi-Function interrupt enable bit of the interrupt control register to enable the SIM interrupt and Multi-function interrupt.



I²C Bus Initialisation Flow Chart

I²C Bus Start Signal

The START signal can only be generated by the master device connected to the I²C bus and not by the slave device. This START signal will be detected by all devices connected to the I²C bus. When detected, this indicates that the I²C bus is busy and therefore the HBB bit will be set. A START condition occurs when a high to low transition on the SDA line takes place when the SCL line remains high.



Start Bit

Slave Address

The transmission of a START signal by the master will be detected by all devices on the I²C bus. To determine which slave device the master wishes to communicate with, the address of the slave device will be sent out immediately following the START signal. All slave devices, after receiving this 7-bit address data, will compare it with their own 7-bit slave address. If the address sent out by the master matches the internal address of the microcontroller slave device, then an internal I²C bus interrupt signal will be generated. The next bit following the address, which is the 8th bit, defines the read/write status and will be saved to the SRW bit of the SIMC1 register. The slave device will then transmit an acknowledge bit, which is a low level, as the 9th bit. The slave device will also set the status flag HAAS when the addresses match.

As an I²C bus interrupt can come from two sources, when the program enters the interrupt subroutine, the HAAS bit should be examined to see whether the interrupt source has come from a matching slave address or from the completion of a data byte transfer. When a slave address is matched, the device must be placed in either the transmit mode and then write data to the SIMD register, or in the receive mode where it must implement a dummy read from the SIMD register to release the SCL line.

I²C Bus Read/Write Signal

The SRW bit in the SIMC1 register defines whether the slave device wishes to read data from the I²C bus or write data to the I²C bus. The slave device should examine this bit to determine if it is to be a transmitter or a receiver. If the SRW flag is 1, then this indicates that the master device wishes to read data from the I²C bus, therefore the slave device must be setup to send data to the I²C bus as a transmitter. If the SRW flag is 0, then this indicates that the master wishes to send data to the I²C bus, therefore the slave device must be setup to read data from the I²C bus as a receiver.

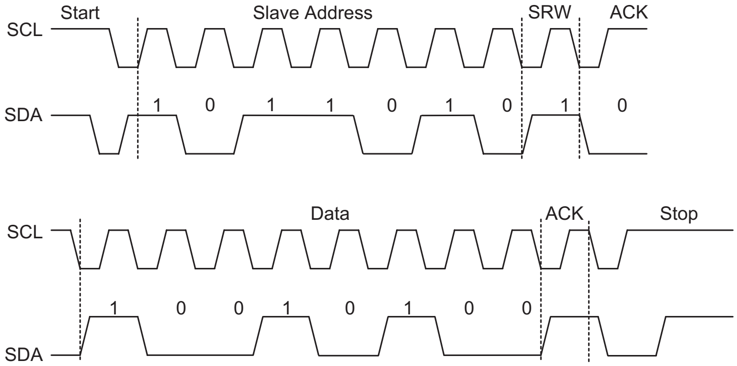
I²C Bus Slave Address Acknowledge Signal

After the master has transmitted a calling address, any slave device on the I²C bus, whose own internal address matches the calling address, must generate an acknowledge signal. The acknowledge signal will inform the master that a slave device has accepted its calling address. If no acknowledge signal is received by the master then a STOP signal must be transmitted by the master to end the communication. When the HAAS flag is high, the addresses have matched and the slave device must check the SRW flag to determine if it is to be a transmitter or a receiver. If the SRW flag is high, the slave device should be setup to be a transmitter so the HTX bit in the SIMC1 register should be set to 1. If the SRW flag is low, then the microcontroller slave device should be setup as a receiver and the HTX bit in the SIMC1 register should be set to 0.

I^{2}C Bus Data and Acknowledge Signal

The transmitted data is 8-bits wide and is transmitted after the slave device has acknowledged receipt of its slave address. The order of serial bit transmission is the MSB first and the LSB last. After receipt of 8-bits of data, the receiver must transmit an acknowledge signal, level 0, before it can receive the next data byte. If the slave transmitter does not receive an acknowledge bit signal from the master receiver, then the slave transmitter will release the SDA line to allow the master to send a STOP signal to release the I^{2}C Bus. The corresponding data will be stored in the SIMD register. If setup as a transmitter, the slave device must first write the data to be transmitted into the SIMD register. If setup as a receiver, the slave device must read the transmitted data from the SIMD register.

When the slave receiver receives the data byte, it must generate an acknowledge bit, known as TXAK, on the 9th clock. The slave device, which is setup as a transmitter will check the RXAK bit in the SIMC1 register to determine if it is to send another data byte, if not then it will release the SDA line and await the receipt of a STOP signal from the master.

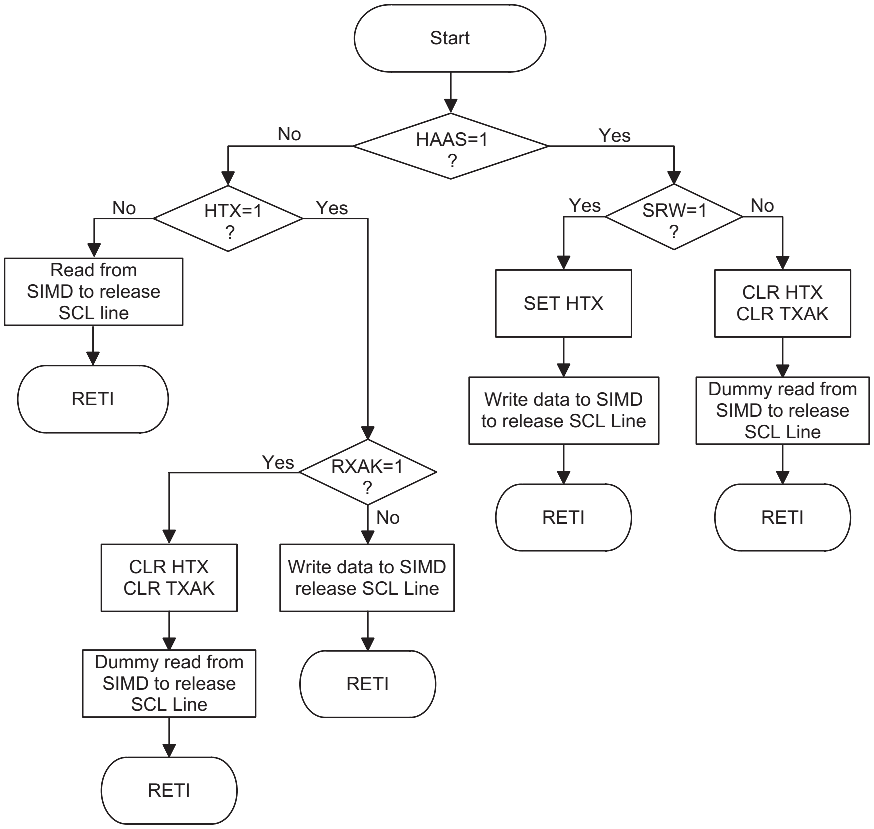


S=Start (1 bit)
SA=Slave Address (7 bits)
SR=SRW bit (1 bit)
M=Slave device send acknowledge bit (1 bit)
D=Data (8 bits)
A=ACK (RXAK bit for transmitter, TXAK bit for receiver 1 bit)
P=Stop (1 bit)

S	SA	SR	M	D	A	D	A	--------	S	SA	SR	M	D	A	D	A	--------	P

Note: *When a slave address is matched, the devices must be placed in either the transmit mode and then write data to the SIMD register, or in the receive mode where it must implement a dummy read from the SIMD register to release the SCL line.

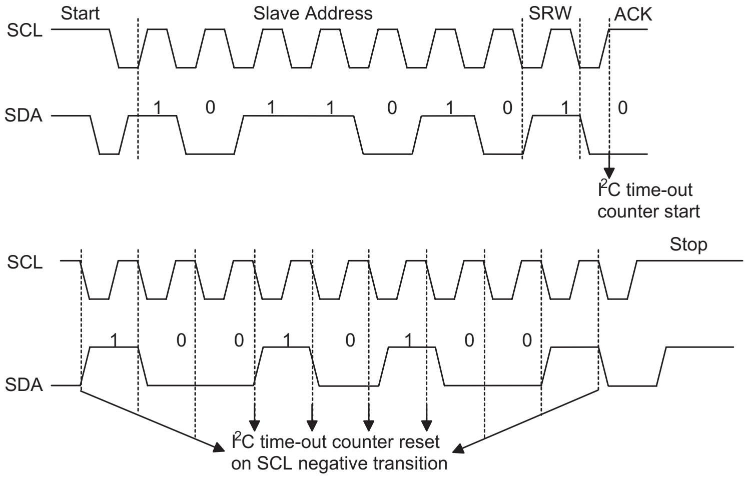
I^{2}C Communication Timing Diagram


Start
HAAS=1 ?
No
Yes
HTX=1 ?
No
Yes
Read from SIMD to release SCL line
RETI
RXAK=1 ?
Yes
No
CLR HTX CLR TXAK
Dummy read from SIMD to release SCL Line
RETI
Write data to SIMD release SCL Line
RETI
SRW=1 ?
Yes
No
SET HTX
Write data to SIMD to release SCL Line
RETI
CLR HTX CLR TXAK
Dummy read from SIMD to release SCL Line
RETI

I^{2}C Bus ISR Flow Chart

I^{2}C Time-out Control

In order to reduce the I^{2}C lockup problem due to reception of erroneous clock sources, a time-out function is provided. If the clock source connected to the I^{2}C bus is not received for a while, then the I^{2}C circuitry and registers will be reset after a certain time-out period. The time-out counter starts to count on an I^{2}C bus "START" & "address match" condition, and is cleared by an SCL falling edge. Before the next SCL falling edge arrives, if the time elapsed is greater than the time-out period specified by the I2CTOC register, then a time-out condition will occur. The time-out function will stop when an I^{2}C "STOP" condition occurs.


Start
Slave Address
SRW
ACK
SCL
SDA
1
0
1
1
0
1
0
1
0
I²C time-out counter start
SCL
Stop
SDA
1
0
0
1
0
1
0
0
I²C time-out counter reset on SCL negative transition

I²C Time-out

When an I²C time-out counter overflow occurs, the counter will stop and the I2CTOEN bit will be cleared to zero and the I2CTF bit will be set high to indicate that a time-out condition has occurred. The time-out condition will also generate an interrupt which uses the I²C interrupt vector. When an I²C time-out occurs, the I²C internal circuitry will be reset and the registers will be reset into the following condition:

Register	After I²C Time-out
SIMD, SIMA, SIMC0	No change
SIMC1	Reset to POR condition

I²C Registers after Time-out

The I2CTOF flag can be cleared by the application program. There are 64 time-out period selections which can be selected using the I2CTOS bits in the I2CTOC register. The time-out duration is calculated by the formula: $((1\sim64)\times(32/f_{SUB}))$. This gives a time-out period which ranges from about 1ms to 64ms.

I2CTOC Register

Bit	7	6	5	4	3	2	1	0
Name	I2CTOEN	I2CTOF	I2CTOS5	I2CTOS4	I2CTOS3	I2CTOS2	I2CTOS1	I2CTOS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **I2CTOEN:** I²C Time-out Control
0: Disable
1: Enable

Bit 6 **I2CTOF:** I²C Time-out flag
0: No time-out occurred
1: time-out occurred

Bit 5~0 **I2CTOS5~I2CTOS0:** I²C Time-out Time Selection
I²C Time-out clock source is $f_{SUB}/32$
I²C Time-out time is given by: $(\text{I2CTOS}[5:0]+1)\times(32/f_{SUB})$

Peripheral Clock Output

The Peripheral Clock Output allows the devices to supply external hardware with a clock signal synchronised to the microcontroller clock.

Peripheral Clock Operation

As the peripheral clock output pin, PCK, is shared with I/O line, the required pin function is chosen via PCKEN in the SIMC0 register. The Peripheral Clock function is controlled using the SIMC0 register. The clock source for the Peripheral Clock Output can originate from either the TM0 CCRP match frequency/2 or a divided ratio of the internal f_{SYS} clock. The PCKEN bit in the SIMC0 register is the overall on/off control, setting PCKEN bit to 1 enables the Peripheral Clock and setting PCKEN bit to 0 disables it. The required division ratio of the system clock is selected using the PCKP1 and PCKP0 bits in the same register. If the devices enter the SLEEP Mode, this will disable the Peripheral Clock output.

SIMC0 Register

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	PCKEN	PCKP1	PCKP0	SIMEN	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	—
POR	1	1	1	0	0	0	0	—

Bit 7~5 **SIM2, SIM1, SIM0:** SIM Operating Mode Control
000: SPI master mode; SPI clock is $f_{SYS}/4$
001: SPI master mode; SPI clock is $f_{SYS}/16$
010: SPI master mode; SPI clock is $f_{SYS}/64$
011: SPI master mode; SPI clock is f_{SUB}
100: SPI master mode; SPI clock is TM0 CCRP match frequency/2
101: SPI slave mode
110: I^2C slave mode
111: Unused mode

These bits setup the overall operating mode of the SIM function. As well as selecting if the I^2C or SPI function, they are used to control the SPI Master/Slave selection and the SPI Master clock frequency. The SPI clock is a function of the system clock but can also be chosen to be sourced from the TM0. If the SPI Slave Mode is selected then the clock will be supplied by an external Master device.

Bit 4 **PCKEN:** PCK Output Pin Control
0: Disable
1: Enable

Bit 3~2 **PCKP1, PCKP0**: Select PCK output pin frenquency
00: f_{SYS}
01: $f_{SYS}/4$
10: $f_{SYS}/8$
11: TM0 CCRP match frequency/2

Bit 1 **SIMEN:** SIM Control
0: Disable
1: Enable

The bit is the overall on/off control for the SIM interface. When the SIMEN bit is cleared to zero to disable the SIM interface, the SDI, SDO, SCK and $\overline{\text{SCS}}$, or SDA and SCL lines will be in a floating condition and the SIM operating current will be reduced to a minimum value. When the bit is high the SIM interface is enabled. The SIM configuration option must have first enabled the SIM interface for this bit to be effective. If the SIM is configured to operate as an SPI interface via the SIM2~SIM0 bits, the contents of the SPI control registers will remain at the previous settings when the SIMEN bit changes from low to high and should therefore be first initialised by the application program. If the SIM is configured to operate as an I^{2}C interface via the SIM2~SIM0 bits and the SIMEN bit changes from low to high, the contents of the I^{2}C control bits such as HTX and TXAK will remain at the previous settings and should therefore be first initialised by the application program while the relevant I^{2}C flags such as HCF, HAAS, HBB, SRW and RXAK will be set to their default states.

Bit 0 Unimplemented, read as "0"

Serial Interface – SPIA

The devices contain an independent SPI function. It is important not to confuse this independent SPI function with the additional one contained within the combined SIM function, which is described in another section of this datasheet. This independent SPI function will carry the name SPIA to distinguish it from the other one in the SIM.

This SPIA interface is often used to communicate with external peripheral devices such as sensors, Flash or EEPROM memory devices, etc. Originally developed by Motorola, the four-line SPI interface is a synchronous serial data interface that has a relatively simple communication protocol simplifying the programming requirements when communicating with external hardware devices.

The communication is full duplex and operates as a slave/master type, where the devices can be either master or slave. Although the SPIA interface specification can control multiple slave devices from a single master, these devices are provided only one $\overline{\text{SCSA}}$ line. If the master needs to control multiple slave devices from a single master, the master can use I/O pins to select the slave devices.

SPIA Interface Operation

The SPIA interface is a full duplex synchronous serial data link. It is a four line interface with pin names SDIA, SDOA, SCKA and $\overline{\text{SCSA}}$. Signals SDIA and SDOA are the Serial Data Input and Serial Data Output lines, SCKA is the Serial Clock line and $\overline{\text{SCSA}}$ is the Slave Select line. As the SPIA interface lines are pin-shared with other functions, the SPIA interface lines must first be enabled by selecting the SPIA interface enable configuration option and setting the correct bits in the SPIAC0 and SPIAC1 registers. After the SPIA enable configuration option has been configured, the SPIA interface function can also be additionally disabled or enabled using the SPIAEN bit in the SPIAC0 register. Communication between devices connected to the SPIA interface is carried out in a slave/master mode with all data transfer initiations being implemented by the master. The master also controls the clock/signal. As the devices only contain a single $\overline{\text{SCSA}}$ pin only one slave device can be utilised.

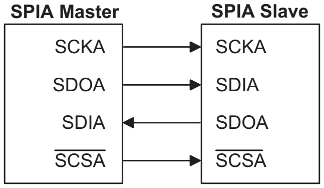
The $\overline{\text{SCSA}}$ line is controlled by the application program, set the SACSEN bit to “1” to enable the $\overline{\text{SCSA}}$ line function and clear the SACSEN bit to “0” to place the $\overline{\text{SCSA}}$ pin into other pin-shared functions.

The SPIA Serial Interface function includes the following features:

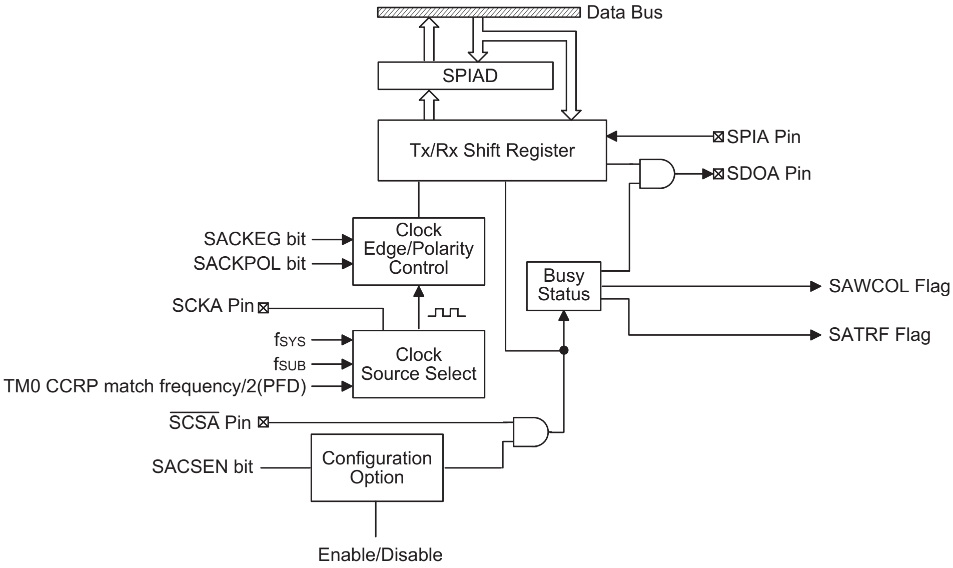
- Full duplex synchronous data transfer
- Both Master and Slave modes
- LSB first or MSB first data transmission modes
- Transmission complete flag
- Rising or falling active clock edge
- SAWCOL and SACSEN bit enabled or disable select

The status of the SPIA interface lines is determined by a number of factors such as whether the devices are in the master or slave mode and upon the condition of certain control bits such as SACSEN and SPIAEN.

There are several configuration options associated with the SPIA interface. One of these is to enable the SPIA function which selects the SPIA lines rather than normal I/O pins. Note that if the configuration option does not select the SPIA function then the SPIAEN bit in the SPIAC0 register will have no effect. Another two SPIA configuration options determine if the SACSEN and SAWCOL bits are to be used.



SPIA Master/Slave Connection



SPIA Block Diagram

SPIA Registers

There are three internal registers which control the overall operation of the SPIA interface. These are the SPIAD data register and two registers SPIAC0 and SPIAC1.

Register Name	Bit							
	7	6	5	4	3	2	1	0
SPIAC0	SASPI2	SASPI1	SASPI0	—	—	—	SPIAEN	—
SPIAC1	—	—	SACKPOL	SACKEG	SAMLS	SACSEN	SAWCOL	SATRF
SPIAD	D7	D6	D5	D4	D3	D2	D1	D0

SPIA Registers List

The SPIAD register is used to store the data being transmitted and received. Before the devices write data to the SPIA bus, the actual data to be transmitted must be placed in the SPIAD register. After the data is received from the SPIA bus, the devices can read it from the SPIAD register. Any transmission or reception of data from the SPIA bus must be made via the SPIAD register.

SPIAD Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	×	×	×	×	×	×	×	×

"×" unknown

There are also two control registers for the SPIA interface, SPIAC0 and SPIAC1. Register SPIAC0 is used to control the enable or disable function and to set the data transmission clock frequency. Register SPIAC1 is used for other control functions such as LSB/MSB selection, write collision flag, etc.

SPIAC0 Register

Bit	7	6	5	4	3	2	1	0
Name	SASPI2	SASPI1	SASPI0	—	—	—	SPIAEN	—
R/W	R/W	R/W	R/W	—	—	—	R/W	—
POR	1	1	1	—	—	—	0	—

Bit 7~5 **SASPI2~SASPI0:** SPIA Master/Slave Clock Select
000: SPIA master, $f_{SYS}/4$
001: SPIA master, $f_{SYS}/16$
010: SPIA master, $f_{SYS}/64$
011: SPIA master, f_{SUB}
100: SPIA master, TM0 CCRP match frequency/2 (PFD)
101: SPIA slave
11x: Reserved

Bit 4~2 Unimplemented, read as "0"

Bit 1 **SPIAEN:** SPIA enable or disable
0: Disable
1: Enable

The bit is the overall on/off control for the SPIA interface. When the SPIAEN bit is cleared to zero to disable the SPIA interface, the SDIA, SDOA, SCKA and $\overline{SCSA}$ lines will lose their SPI function and the SPIA operating current will be reduced to a minimum value. When the bit is high, the SPIA interface is enabled.

Bit 0 Unimplemented, read as "0"

SPIAC1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	SACKPOL	SACKEG	SAMLS	SACSEN	SAWCOL	SATRF
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 Unimplemented, read as “0”

Bit 5 **SACKPOL:** Determines the base condition of the clock line
0: SCKA line will be high when the clock is inactive
1: SCKA line will be low when the clock is inactive

The SACKPOL bit determines the base condition of the clock line, if the bit is high, then the SCKA line will be low when the clock is inactive. When the SACKPOL bit is low, then the SCKA line will be high when the clock is inactive.

Bit 4 **SACKEG:** Determines the SPIA SCKA active clock edge type
SACKPOL=0:
0: SCKA has high base level with data capture on SCKA rising edge
1: SCKA has high base level with data capture on SCKA falling edge
SACKPOL=1:
0: SCKA has low base level with data capture on SCKA falling edge
1: SCKA has low base level with data capture on SCKA rising edge

The SACKEG and SACKPOL bits are used to setup the way that the clock signal outputs and inputs data on the SPIA bus. These two bits must be configured before a data transfer is executed otherwise an erroneous clock edge may be generated. The SACKPOL bit determines the base condition of the clock line, if the bit is high, then the SCKA line will be low when the clock is inactive. When the SACKPOL bit is low, then the SCKA line will be high when the clock is inactive. The SACKEG bit determines active clock edge type which depends upon the condition of the SACKPOL bit.

Bit 3 **SAMLS:** data shift order
0: the LSB of data is transmitted first
1: the MSB of data is transmitted first

Bit 2 **SACSEN:** SPIA $\overline{\text{SCSA}}$ pin Control
0: Disable
1: Enable

The SACSEN bit is used as an enable/disable for the $\overline{\text{SCSA}}$ pin. If this bit is low, then the $\overline{\text{SCSA}}$ pin function will be disabled and used as an I/O function. If the bit is high the $\overline{\text{SCSA}}$ pin will be enabled and used as a select pin. Note that using the SACSEN bit can be disabled or enabled via a configuration option.

Bit 1 **SAWCOL:** SPIA Write Collision flag
0: Collision free
1: Collision detected

The SAWCOL flag is used to detect if a data collision has occurred. If this bit is high it means that data has been attempted to be written to the SPIAD register during a data transfer operation. This writing operation will be ignored if data is being transferred. The bit can be cleared by the application program. Note that using the SAWCOL bit can be disabled or enabled via a configuration option.

Bit 0 **SATRF:** SPIA Transmit/Receive Complete flag
0: Data is being transferred
1: SPIA data transmission is completed

The SATRF bit is the Transmit/Receive Complete flag and is set to “1” automatically when an SPIA data transmission is completed, but must cleared to “0” by the application program. It can be used to generate an interrupt.

SPIA Communication

After the SPIA interface is enabled by setting the SPIAEN bit high, then in the Master Mode, when data is written to the SPIAD register, transmission/reception will begin simultaneously. When the data transfer is complete, the SATRF flag will be set automatically, but must be cleared using the application program. In the Slave Mode, when the clock signal from the master has been received, any data in the SPIAD register will be transmitted and any data on the SDIA pin will be shifted into the SPIAD registers.

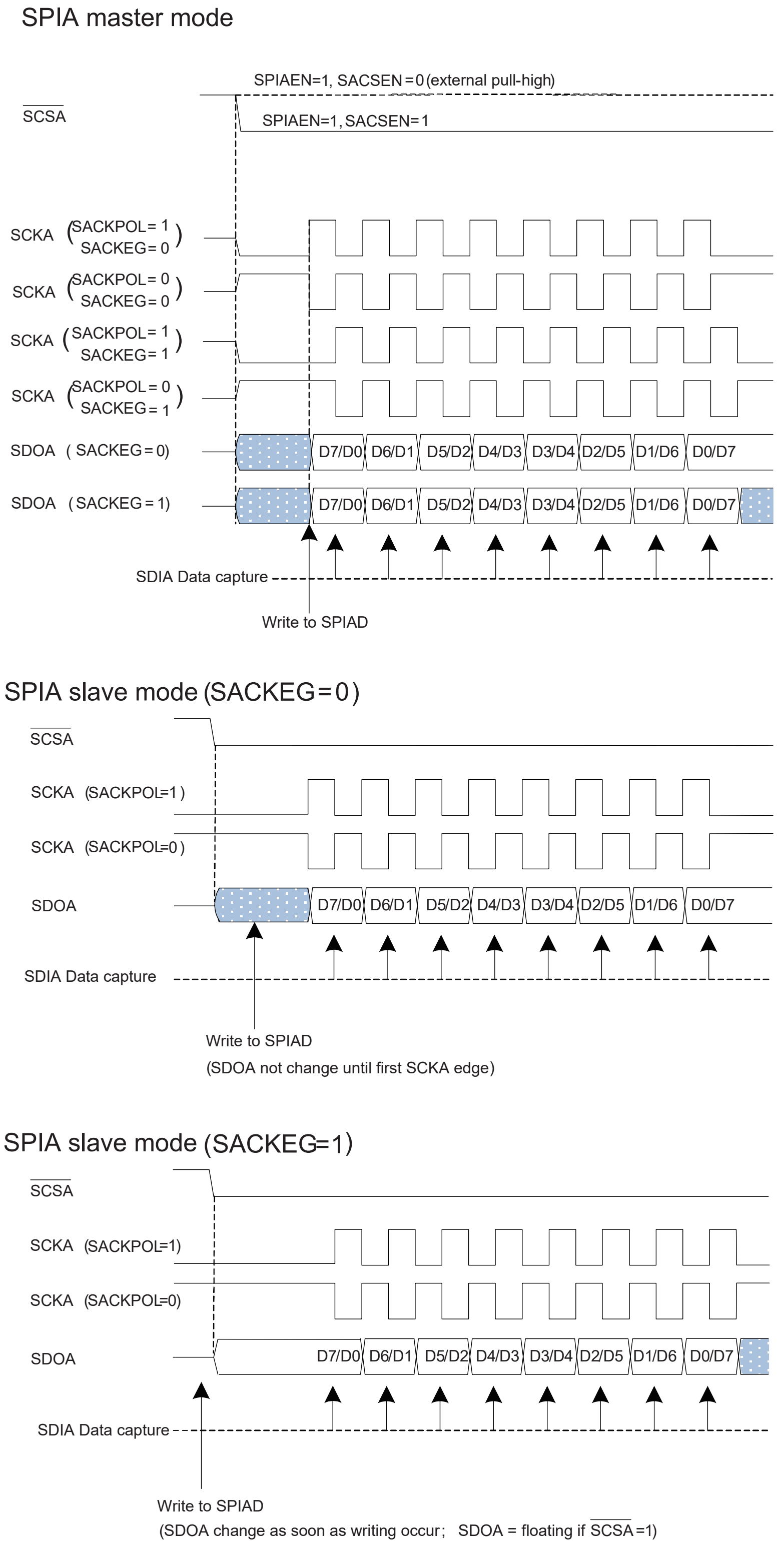
The master should output a $\overline{\text{SCSA}}$ signal to enable the slave device before a clock signal is provided. The slave data to be transferred should be well prepared at the appropriate moment relative to the $\overline{\text{SCSA}}$ signal depending upon the configurations of the SACKPOL bit and SACKEG bit. The accompanying timing diagram shows the relationship between the slave data and $\overline{\text{SCSA}}$ signal for various configurations of the SACKPOL and SACKEG bits.

The SPIA will continue to function even in the IDLE Mode.

SPIA Bus Enable/Disable

To enable the SPIA bus, set SACSEN=1 and $\overline{\text{SCSA}}$=0, then wait for data to be written into the SPIAD (TXRX buffer) register. For the Master Mode, after data has been written to the SPIAD (TXRX buffer) register, then transmission or reception will start automatically. When all the data has been transferred the SATRF bit should be set. For the Slave Mode, when clock pulses are received on SCKA, data in the TXRX buffer will be shifted out or data on SDIA will be shifted in.

To disable the SPIA bus the SCKA, SDIA, SDOA and $\overline{\text{SCSA}}$ lines will become I/O pins or other pin-shared functions.


SPIA master mode
SPIAEN=1, SACSEN=0 (external pull-high)
SCSA
SPIAEN=1, SACSEN=1
SCKA (SACKPOL=1 SACKEG=0)
SCKA (SACKPOL=0 SACKEG=0)
SCKA (SACKPOL=1 SACKEG=1)
SCKA (SACKPOL=0 SACKEG=1)
SDOA (SACKEG=0)
SDOA (SACKEG=1)
D7/D0 D6/D1 D5/D2 D4/D3 D3/D4 D2/D5 D1/D6 D0/D7
SDIA Data capture
Write to SPIAD
SPIA slave mode (SACKEG=0)
SCSA
SCKA (SACKPOL=1)
SCKA (SACKPOL=0)
SDOA
D7/D0 D6/D1 D5/D2 D4/D3 D3/D4 D2/D5 D1/D6 D0/D7
SDIA Data capture
Write to SPIAD
(SDOA not change until first SCKA edge)
SPIA slave mode (SACKEG=1)
SCSA
SCKA (SACKPOL=1)
SCKA (SACKPOL=0)
SDOA
D7/D0 D6/D1 D5/D2 D4/D3 D3/D4 D2/D5 D1/D6 D0/D7
SDIA Data capture
Write to SPIAD
(SDOA change as soon as writing occur; SDOA = floating if SCSA=1)

Note: For SPIA slave mode, if SPIAEN=1 and SACSEN=0, SPIA is always enabled and ignore the $\overline{\text{SCSA}}$ level.

SPIA Master/Slave Mode Timing Diagram

SPIA Operation

All communication is carried out using the 4-line interface for either Master or Slave Mode.

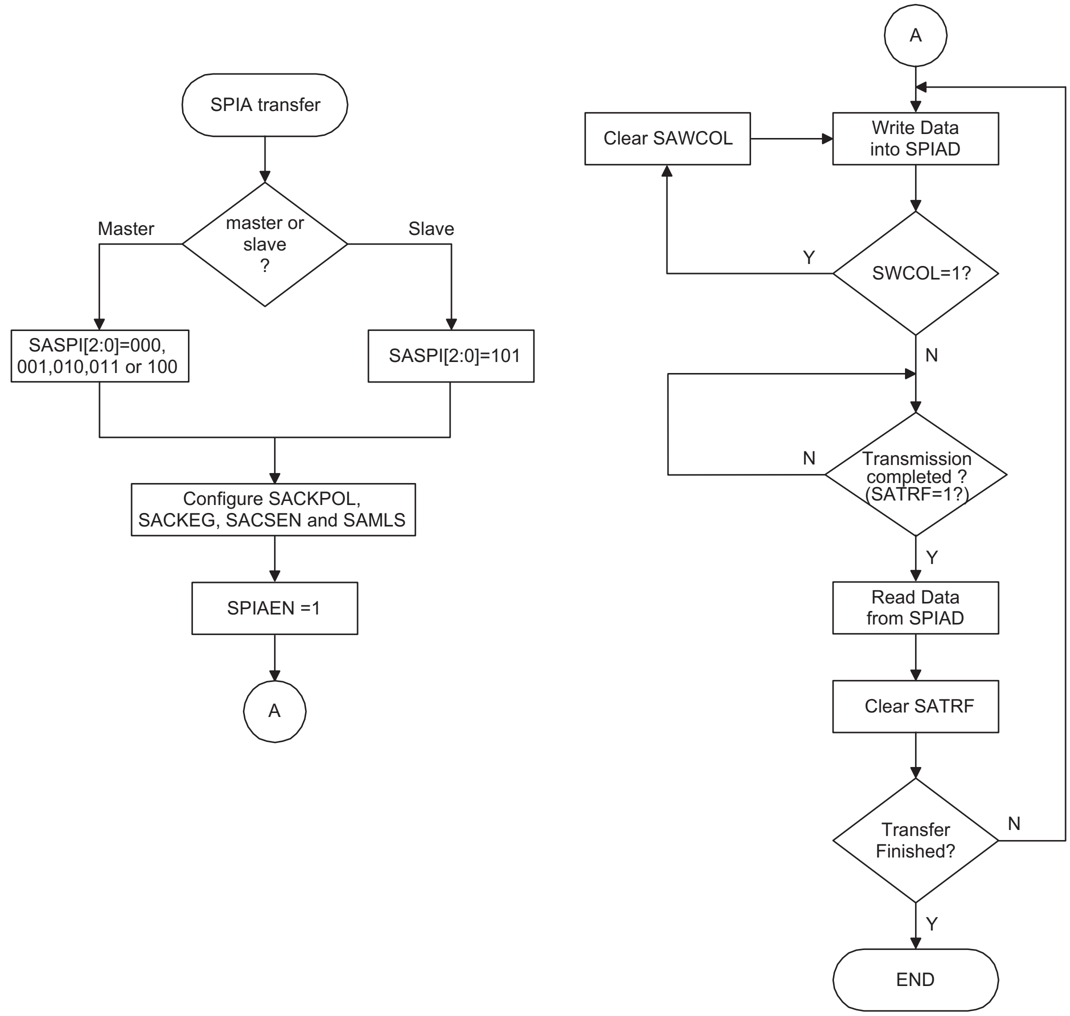
The SACSEN bit in the SPIAC1 register controls the overall function of the SPIA interface. Setting this bit high will enable the SPIA interface by allowing the $\overline{\text{SCSA}}$ line to be active, which can then be used to control the SPIA interface. If the SACSEN bit is low, the SPIA interface will be disabled and the $\overline{\text{SCSA}}$ line will be an I/O pin or other pin-shared functions and can therefore not be used for control of the SPIA interface. If the SACSEN bit and the SPIAEN bit in the SPIAC0 register are set high, this will place the SDIA line in a floating condition and the SDOA line high. If in Master Mode the SCKA line will be either high or low depending upon the clock polarity selection bit SACKPOL in the SPIAC1 register. If in Slave Mode the SCKA line will be in a floating condition. If SPIAEN is low then the bus will be disabled and $\overline{\text{SCSA}}$, SDIA, SDOA and SCKA lines will all become I/O pins or other pin-shared functions. In the Master Mode the Master will always generate the clock signal. The clock and data transmission will be initiated after data has been written into the SPIAD register. In the Slave Mode, the clock signal will be received from an external master device for both data transmission and reception. The following sequences show the order to be followed for data transfer in both Master and Slave Mode:

Master Mode

- Step 1
  Select the clock source and Master mode using the SASPI2~SASPI0 bits in the SPIAC0 control register
- Step 2
  Setup the SACSEN bit and setup the SAMLS bit to choose if the data is MSB or LSB shifted first, this must be same as the Slave device.
- Step 3
  Setup the SPIAEN bit in the SPIAC0 control register to enable the SPIA interface.
- Step 4
  For write operations: write the data to the SPIAD register, which will actually place the data into the TXRX buffer. Then use the SCKA and $\overline{\text{SCSA}}$ lines to output the data. After this go to step 5.
  For read operations: the data transferred in on the SDIA line will be stored in the TXRX buffer until all the data has been received at which point it will be latched into the SPIAD register.
- Step 5
  Check the SAWCOL bit if set high then a collision error has occurred so return to step 4. If equal to zero then go to the following step.
- Step 6
  Check the SATRF bit or wait for a SPIA serial bus interrupt.
- Step 7
  Read data from the SPIAD register.
- Step 8
  Clear SATRF.
- Step 9
  Go to step 4.

Slave Mode

- Step 1
  Select the SPI Slave mode using the SASPI2~SASPI0 bits in the SPIAC0 control register
- Step 2
  Setup the SACSEN bit and setup the SAMLS bit to choose if the data is MSB or LSB shifted first, this setting must be the same with the Master device.
- Step 3
  Setup the SPIAEN bit in the SPIAC0 control register to enable the SPIA interface.
- Step 4
  For write operations: write the data to the SPIAD register, which will actually place the data into the TXRX buffer. Then wait for the master clock SCKA and $\overline{\text{SCSA}}$ signal. After this, go to step 5.
  For read operations: the data transferred in on the SDIA line will be stored in the TXRX buffer until all the data has been received at which point it will be latched into the SPIAD register.
- Step 5
  Check the SAWCOL bit if set high then a collision error has occurred so return to step 4. If equal to zero then go to the following step.
- Step 6
  Check the SATRF bit or wait for a SPIA serial bus interrupt.
- Step 7
  Read data from the SPIAD register.
- Step 8
  Clear SATRF.
- Step 9
  Go to step 4.



SPIA Transfer Control Flowchart

Error Detection

The SAWCOL bit in the SPIAC1 register is provided to indicate errors during data transfer. The bit is set by the SPIA serial Interface but must be cleared by the application program. This bit indicates a data collision has occurred which happens if a write to the SPIAD register takes place during a data transfer operation and will prevent the write operation from continuing.

Interrupts

Interrupts are an important part of any microcontroller system. When an external event or an internal function such as a Timer Module or an A/D converter requires microcontroller attention, their corresponding interrupt will enforce a temporary suspension of the main program allowing the microcontroller to direct attention to their respective needs. The devices contain several external interrupt and internal interrupts functions. The external interrupts are generated by the action of the external INT0 and INT1 pins while the internal interrupts are generated by various internal functions such as the TMs, Time Base, LVD and the A/D converter, etc.

Interrupt Registers

Overall interrupt control, which basically means the setting of request flags when certain microcontroller conditions occur and the setting of interrupt enable bits by the application program, is controlled by a series of registers, located in the Special Purpose Data Memory, as shown in the accompanying table. The number of registers depends upon the devices chosen but fall into three categories. The first is the INTC0~INTC2 registers which setup the primary interrupts, the second is the MFI0~MFI4 registers which setup the Multi-function interrupts. Finally there is an INTEG register to setup the external interrupt trigger edge type.

Each register contains a number of enable bits to enable or disable individual registers as well as interrupt flags to indicate the presence of an interrupt request. The naming convention of these follows a specific pattern. First is listed an abbreviated interrupt type, then the (optional) number of that interrupt followed by either an ”E” for enable/disable bit or “F” for request flag.

Function	Enable Bit	Request Flag	Notes
Global	EMI	—	—
INTn Pins	INTnE	INTnF	n=0 or 1
Comparator	CPE	CPF	—
Multi-function	MFnE	MFnF	n=0~4
A/D converter	ADE	ADF	—
Time Base	TBnE	TBnF	n=0 or 1
LVD	LVE	LVF	—
EEPROM	DEE	DEF	—
SIM	SIME	SIMF	—
SPIA	SPIAE	SPIAF	—
TM	TnPE	TnPF	n=0~3
	TnAE	TnAF	
	TnBE	TnBF	n=3

Interrupt Register Bit Naming Conventions

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	MF0F	CPF	INT0F	MF0E	CPE	INT0E	EMI
INTC1	MF4F	MF3F	MF2F	MF1F	MF4E	MF3E	MF2E	MF1E
INTC2	INT1F	TB1F	TB0F	ADF	INT1E	TB1E	TB0E	ADE
MFI0	—	—	T0AF	T0PF	—	—	T0AE	T0PE
MFI2	—	—	T2AF	T2PF	—	—	T2AE	T2PE
MFI3	—	T3BF	T3AF	T3PF	—	T3BE	T3AE	T3PE
MFI4	SPIAF	SIMF	DEF	LVF	SPIAE	SIME	DEE	LVE

BC66F840 Interrupt Register List

Register Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	MF0F	CPF	INT0F	MF0E	CPE	INT0E	EMI
INTC1	MF4F	MF3F	MF2F	MF1F	MF4E	MF3E	MF2E	MF1E
INTC2	INT1F	TB1F	TB0F	ADF	INT1E	TB1E	TB0E	ADE
MFI0	—	—	T0AF	T0PF	—	—	T0AE	T0PE
MFI1	—	—	T1AF	T1PF	—	—	T1AE	T1PE
MFI2	—	—	T2AF	T2PF	—	—	T2AE	T2PE
MFI3	—	T3BF	T3AF	T3PF	—	T3BE	T3AE	T3PE
MFI4	SPIAF	SIMF	DEF	LVF	SPIAE	SIME	DEE	LVE

BC66F850/BC66F860 Interrupt Register List

INTEG Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4 Unimplemented, read as “0”

Bit 3~2 **INT1S1~INT1S0:** interrupt edge control for INT1 pin
00: Disable
01: Rising edge
10: Falling edge
11: Both rising and falling edges

Bit 1~0 **INT0S1~INT0S0:** interrupt edge control for INT0 pin
00: Disable
01: Rising edge
10: Falling edge
11: Both rising and falling edges

INTC0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	MF0F	CPF	INT0F	MF0E	CPE	INT0E	EMI
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7 Unimplemented, read as “0”

Bit 6 **MF0F:** Multi-function 0 Interrupt request flag
0: No request
1: Interrupt request

Bit 5 **CPF:** Comparator interrupt request flag
0: No request
1: Interrupt request

Bit 4 **INT0F:** INT0 interrupt request flag
0: No request
1: Interrupt request

Bit 3 **MF0E:** Multi-function 0 Interrupt control
0: Disable
1: Enable

Bit 2 **CPE:** Comparator Interrupt control
0: Disable
1: Enable

Bit 1 **INT0E:** INT0 interrupt control
0: Disable
1: Enable

Bit 0 **EMI:** Global interrupt control
0: Disable
1: Enable

INTC1 Register

Bit	7	6	5	4	3	2	1	0
Name	MF4F	MF3F	MF2F	MF1F	MF4E	MF3E	MF2E	MF1E
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **MF4F:** Multi-function 4 Interrupt request flag
0: No request
1: Interrupt request

Bit 6 **MF3F:** Multi-function 3 Interrupt request flag
0: No request
1: Interrupt request

Bit 5 **MF2F:** Multi-function 2 Interrupt request flag
0: No request
1: Interrupt request

Bit 4 **MF1F:** Multi-function 1 Interrupt request flag
0: No request
1: Interrupt request

Bit 3 **MF4E:** Multi-function 4 Interrupt control
0: Disable
1: Enable

Bit 2 **MF3E:** Multi-function 3 Interrupt control
0: Disable
1: Enable

Bit 1 **MF2E:** Multi-function 2 Interrupt control
0: Disable
1: Enable

Bit 0 **MF1E:** Multi-function 1 Interrupt control
0: Disable
1: Enable

INTC2 Register

Bit	7	6	5	4	3	2	1	0
Name	INT1F	TB1F	TB0F	ADF	INT1E	TB1E	TB0E	ADE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **INT1F:** INT1 interrupt request flag
0: No request
1: Interrupt request

Bit 6 **TB1F:** Time Base 1 Interrupt request flag
0: No request
1: Interrupt request

Bit 5 **TB0F:** Time Base 0 Interrupt request flag
0: No request
1: Interrupt request

Bit 4 **ADF:** A/D Converter interrupt request flag
0: No request
1: Interrupt request

Bit 3 **INT1E:** INT1 interrupt control
0: Disable
1: Enable

Bit 2 **TB1E:** Time Base 1 interrupt control
0: Disable
1: Enable

Bit 1 **TB0E:** Time Base 0 interrupt control
0: Disable
1: Enable

Bit 0 **ADE:** A/D Converter interrupt control
0: Disable
1: Enable

MFI0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	T0AF	T0PF	—	—	T0AE	T0PE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 Unimplemented, read as “0”

Bit 5 **T0AF:** TM0 Comparator A match Interrupt request flag
0: No request
1: Interrupt request

Bit 4 **T0PF:** TM0 Comparator P match Interrupt request flag
0: No request
1: Interrupt request

Bit 3~2 Unimplemented, read as “0”

Bit 1 **T0AE:** TM0 Comparator A match Interrupt control
0: Disable
1: Enable

Bit 0 **T0PE:** TM0 Comparator P match Interrupt control
0: Disable
1: Enable

MFI1 Register – BC66F850/BC66F860

Bit	7	6	5	4	3	2	1	0
Name	—	—	T1AF	T1PF	—	—	T1AE	T1PE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 Unimplemented, read as “0”

Bit 5 **T1AF:** TM1 Comparator A match Interrupt request flag
0: No request
1: Interrupt request

Bit 4 **T1PF:** TM1 Comparator P match Interrupt request flag
0: No request
1: Interrupt request

Bit 3~2 Unimplemented, read as “0”

Bit 1 **T1AE:** TM1 Comparator A match Interrupt control
0: Disable
1: Enable

Bit 0 **T1PE:** TM1 Comparator P match Interrupt control
0: Disable
1: Enable

MFI2 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	T2AF	T2PF	—	—	T2AE	T2PE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 Unimplemented, read as “0”

Bit 5 **T2AF:** TM2 Comparator A match Interrupt request flag
0: No request
1: Interrupt request

Bit 4 **T2PF:** TM2 Comparator P match Interrupt request flag
0: No request
1: Interrupt request

Bit 3~2 Unimplemented, read as “0”

Bit 1 **T2AE:** TM2 Comparator A match Interrupt control
0: Disable
1: Enable

Bit 0 **T2PE:** TM2 Comparator P match Interrupt control
0: Disable
1: Enable

MFI3 Register

Bit	7	6	5	4	3	2	1	0
Name	—	T3BF	T3AF	T3PF	—	T3BE	T3AE	T3PE
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

Bit 7 Unimplemented, read as “0”

Bit 6 **T3BF:** TM3 Comparator B match Interrupt request flag
0: No request
1: Interrupt request

Bit 5 **T3AF:** TM3 Comparator A match Interrupt request flag
0: No request
1: Interrupt request

Bit 4 **T3PF:** TM3 Comparator P match Interrupt request flag
0: No request
1: Interrupt request

Bit 3 Unimplemented, read as “0”

Bit 2 **T3BE:** TM3 Comparator B match Interrupt control
0: Disable
1: Enable

Bit 1 **T3AE:** TM3 Comparator A match Interrupt control
0: Disable
1: Enable

Bit 0 **T3PE:** TM3 Comparator P match Interrupt control
0: Disable
1: Enable

MFI4 Register

Bit	7	6	5	4	3	2	1	0
Name	SPIAF	SIMF	DEF	LVF	SPIAE	SIME	DEE	LVE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **SPIAF:** SPIA Interrupt request flag
0: No request
1: Interrupt request

Bit 6 **SIMF:** SIM Interrupt request flag
0: No request
1: Interrupt request

Bit 5 **DEF:** Data EEPROM Interrupt request flag
0: No request
1: Interrupt request

Bit 4 **LVF:** LVD Interrupt request flag
0: No request
1: Interrupt request

Bit 3 **SPIAE:** SPIA Interrupt control
0: Disable
1: Enable

Bit 2 **SIME:** SIM Interrupt control
0: Disable
1: Enable

Bit 1 **DEE:** Data EEPROM Interrupt control
0: Disable
1: Enable

Bit 0 **LVE:** LVD interrupt control
0: Disable
1: Enable

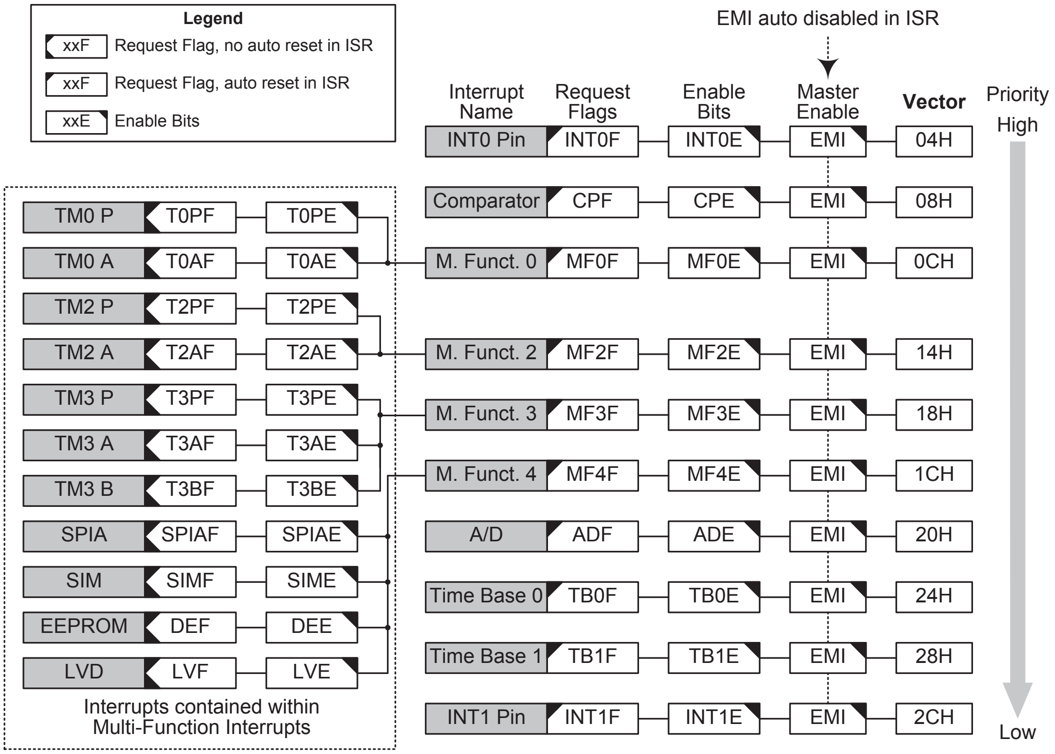
Interrupt Operation

When the conditions for an interrupt event occur, such as a TM Comparator P or Comparator A match or A/D conversion completion etc, the relevant interrupt request flag will be set. Whether the request flag actually generates a program jump to the relevant interrupt vector is determined by the condition of the interrupt enable bit. If the enable bit is set high then the program will jump to its relevant vector; if the enable bit is zero then although the interrupt request flag is set an actual interrupt will not be generated and the program will not jump to the relevant interrupt vector. The global interrupt enable bit, if cleared to zero, will disable all interrupts.

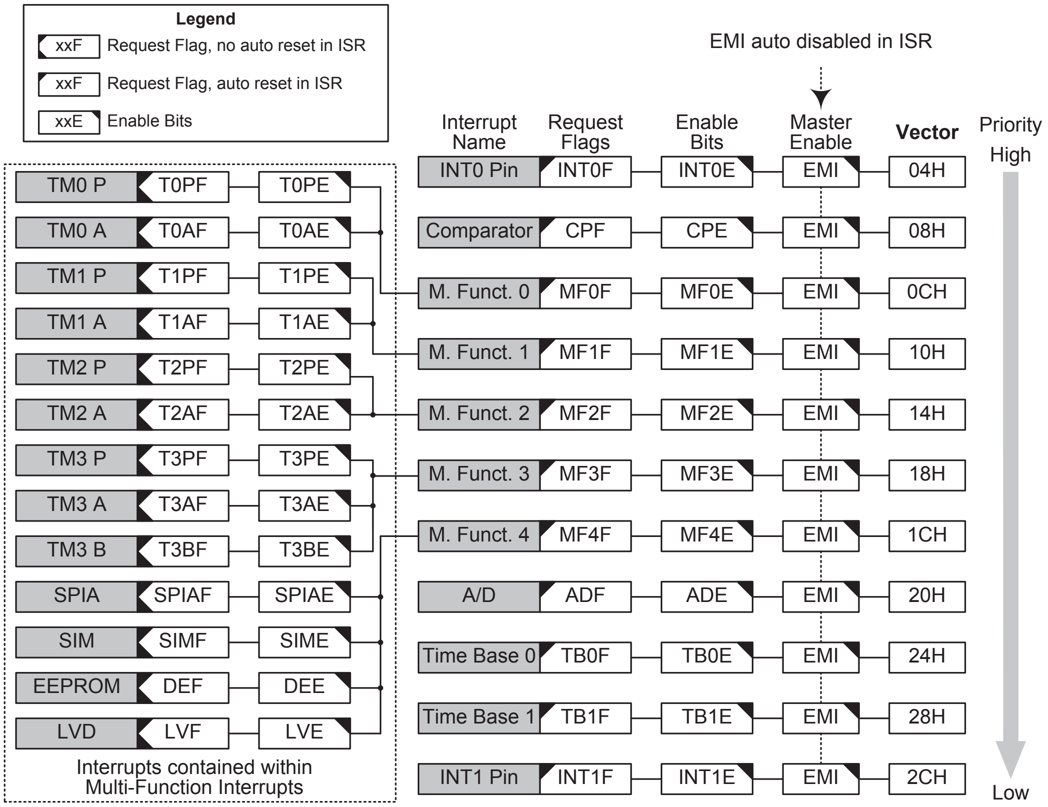
When an interrupt is generated, the Program Counter, which stores the address of the next instruction to be executed, will be transferred onto the stack. The Program Counter will then be loaded with a new address which will be the value of the corresponding interrupt vector. The microcontroller will then fetch its next instruction from this interrupt vector. The instruction at this vector will usually be a JMP which will jump to another section of program which is known as the interrupt service routine. Here is located the code to control the appropriate interrupt. The interrupt service routine must be terminated with a RETI, which retrieves the original Program Counter address from the stack and allows the microcontroller to continue with normal execution at the point where the interrupt occurred.

The various interrupt enable bits, together with their associated request flags, are shown in the accompanying diagrams with their order of priority. Some interrupt sources have their own individual vector while others share the same Multi-function Interrupt vector. Once an interrupt subroutine is serviced, all the other interrupts will be blocked, as the global interrupt enable bit, EMI bit will be cleared automatically. This will prevent any further interrupt nesting from occurring. However, if other interrupt requests occur during this interval, although the interrupt will not be immediately serviced, the request flag will still be recorded.

If an interrupt requires immediate servicing while the program is already in another interrupt service routine, the EMI bit should be set after entering the routine, to allow interrupt nesting. If the stack is full, the interrupt request will not be acknowledged, even if the related interrupt is enabled, until the Stack Pointer is decremented. If immediate service is desired, the stack must be prevented from becoming full. In case of simultaneous requests, the accompanying diagram shows the priority that is applied. All of the interrupt request flags when set will wake-up the devices if it is in SLEEP or IDLE Mode, however to prevent a wake-up from occurring the corresponding flag should be set before the devices are in SLEEP or IDLE Mode.


Legend
xxF Request Flag, no auto reset in ISR
xxF Request Flag, auto reset in ISR
xxE Enable Bits
EMI auto disabled in ISR
Interrupt Name
Request Flags
Enable Bits
Master Enable
Vector
Priority
High
Low
INT0 Pin INT0F INT0E EMI 04H
Comparator CPF CPE EMI 08H
M. Funct. 0 MF0F MF0E EMI 0CH
M. Funct. 2 MF2F MF2E EMI 14H
M. Funct. 3 MF3F MF3E EMI 18H
M. Funct. 4 MF4F MF4E EMI 1CH
A/D ADF ADE EMI 20H
Time Base 0 TB0F TB0E EMI 24H
Time Base 1 TB1F TB1E EMI 28H
INT1 Pin INT1F INT1E EMI 2CH
TM0 P T0PF T0PE
TM0 A T0AF T0AE
TM2 P T2PF T2PE
TM2 A T2AF T2AE
TM3 P T3PF T3PE
TM3 A T3AF T3AE
TM3 B T3BF T3BE
SPIA SPIAF SPIAE
SIM SIMF SIME
EEPROM DEF DEE
LVD LVF LVE
Interrupts contained within Multi-Function Interrupts

Interrupt Structure – BC66F840


Legend
xxF Request Flag, no auto reset in ISR
xxF Request Flag, auto reset in ISR
xxE Enable Bits
EMI auto disabled in ISR
Interrupt Name
Request Flags
Enable Bits
Master Enable
Vector
Priority
High
Low
INT0 Pin INT0F INT0E EMI 04H
Comparator CPF CPE EMI 08H
M. Funct. 0 MF0F MF0E EMI 0CH
M. Funct. 1 MF1F MF1E EMI 10H
M. Funct. 2 MF2F MF2E EMI 14H
M. Funct. 3 MF3F MF3E EMI 18H
M. Funct. 4 MF4F MF4E EMI 1CH
A/D ADF ADE EMI 20H
Time Base 0 TB0F TB0E EMI 24H
Time Base 1 TB1F TB1E EMI 28H
INT1 Pin INT1F INT1E EMI 2CH
TM0 P T0PF T0PE
TM0 A T0AF T0AE
TM1 P T1PF T1PE
TM1 A T1AF T1AE
TM2 P T2PF T2PE
TM2 A T2AF T2AE
TM3 P T3PF T3PE
TM3 A T3AF T3AE
TM3 B T3BF T3BE
SPIA SPIAF SPIAE
SIM SIMF SIME
EEPROM DEF DEE
LVD LVF LVE
Interrupts contained within Multi-Function Interrupts

Interrupt Structure – BC66F850/BC66F860

External Interrupt

The external interrupts are controlled by signal transitions on the INT0~INT1 pins. An external interrupt request will take place when the external interrupt request flags, INT0F~INT1F, are set, which will occur when a transition, whose type is chosen by the edge select bits, appears on the external interrupt pins. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and respective external interrupt enable bit, INT0E~INT1E, must first be set. Additionally the correct interrupt edge type must be selected using the INTEG register to enable the external interrupt function and to choose the trigger edge type. As the external interrupt pins are pin-shared with I/O pins, they can only be configured as external interrupt pins if their external interrupt enable bit in the corresponding interrupt register has been set. The pin must also be setup as an input by setting the corresponding bit in the port control register. When the interrupt is enabled, the stack is not full and the correct transition type appears on the external interrupt pin, a subroutine call to the external interrupt vector, will take place. When the interrupt is serviced, the external interrupt request flags, INT0F~INT1F, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts. Note that any pull-high resistor selections on the external interrupt pins will remain valid even if the pin is used as an external interrupt input.

The INTEG register is used to select the type of active edge that will trigger the external interrupt. A choice of either rising or falling or both edge types can be chosen to trigger an external interrupt. Note that the INTEG register can also be used to disable the external interrupt function. The INT1 pin is internally connected to the 2.4GHz RF Transceiver interrupt output. The correct interrupt edge type must be selected using the INTEG register to ensure the correct interrupt operation.

Comparator Interrupt

The comparator interrupt is controlled by the internal comparator. A comparator interrupt request will take place when the comparator interrupt request flag, CPF, is set, a situation that will occur when the comparator output changes state. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and comparator interrupt enable bit, CPE, must first be set. When the interrupt is enabled, the stack is not full and the comparator inputs generate a comparator output transition, a subroutine call to the comparator interrupt vector, will take place. When the interrupt is serviced, the comparator interrupt request flag, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts.

Multi-function Interrupt

Within these devices there are up to five Multi-function interrupts. Unlike the other independent interrupts, these interrupts have no independent source, but rather are formed from other existing interrupt sources, namely the TM, LVD, EEPROM, SIM and SPIA interrupts.

A Multi-function interrupt request will take place when any of the Multi-function interrupt request flags, MFnF are set. The Multi-function interrupt flags will be set when any of their included functions generate an interrupt request flag. To allow the program to branch to its respective interrupt vector address, when the Multi-function interrupt is enabled and the stack is not full, and either one of the interrupts contained within each of Multi-function interrupt occurs, a subroutine call to one of the Multi-function interrupt vectors will take place. When the interrupt is serviced, the related Multi-Function request flag, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts.

However, it must be noted that, although the Multi-function Interrupt flags will be automatically reset when the interrupt is serviced, the request flags from the original source of the Multi-function interrupts, namely the TM, LVD, EEPROM, SIM and SPIA interrupts, will not be automatically reset and must be manually reset by the application program.

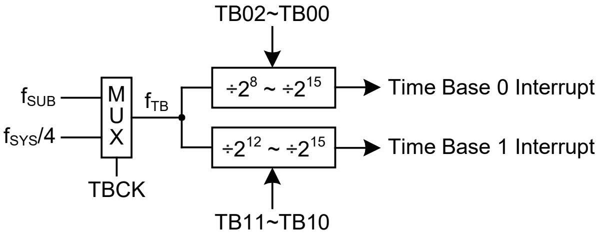
A/D Converter Interrupt

The A/D Converter Interrupt is controlled by the termination of an A/D conversion process. An A/D Converter Interrupt request will take place when the A/D Converter Interrupt request flag, ADF, is set, which occurs when the A/D conversion process finishes. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and A/D Interrupt enable bit, ADE, must first be set. When the interrupt is enabled, the stack is not full and the A/D conversion process has ended, a subroutine call to the A/D Converter Interrupt vector, will take place. When the interrupt is serviced, the A/D Converter Interrupt flag, ADF, will be automatically cleared. The EMI bit will also be automatically cleared to disable other interrupts.

Time Base Interrupt

The function of the Time Base Interrupts is to provide regular time signal in the form of an internal interrupt. They are controlled by the overflow signals from their respective timer functions. When these happens their respective interrupt request flags, TB0F or TB1F will be set. To allow the program to branch to their respective interrupt vector addresses, the global interrupt enable bit, EMI and Time Base enable bits, TB0E or TB1E, must first be set. When the interrupt is enabled, the stack is not full and the Time Base overflows, a subroutine call to their respective vector locations will take place. When the interrupt is serviced, the respective interrupt request flag, TB0F or TB1F, will be automatically reset and the EMI bit will be cleared to disable other interrupts.

The purpose of the Time Base Interrupt is to provide an interrupt signal at fixed time periods. Their clock sources originate from the internal clock source f_{TB}. This f_{TB} input clock passes through a divider, the division ratio of which is selected by programming the appropriate bits in the TBC register to obtain longer interrupt periods whose value ranges. The clock source that generates f_{TB}, which in turn controls the Time Base interrupt period, can originate from several different sources, as shown in the System Operating Mode section.



Time Base Interrupt

TBC Register

Bit	7	6	5	4	3	2	1	0
Name	TBON	TBCK	TB11	TB10	LXTLP	TB02	TB01	TB00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	1	1	0	1	1	1

Bit 7 **TBON:** Time Base control
0: Disable
1: Enable

Bit 6 **TBCK:** Time Base clock f_{TB} selection
0: f_{SUB}
1: $f_{SYS}/4$

Bit 5~4 **TB11~TB10:** Select Time Base 1 Time-out Period
00: $4096/f_{TB}$
01: $8192/f_{TB}$
10: $16384/f_{TB}$
11: $32768/f_{TB}$

Bit 3 **LXTLP:** LXT Oscillator Low Power Control
0: Disable
1: Enable

Bit 2~0 **TB02~TB00:** Select Time Base 0 Time-out Period
000: $256/f_{TB}$
001: $512/f_{TB}$
010: $1024/f_{TB}$
011: $2048/f_{TB}$
100: $4096/f_{TB}$
101: $8192/f_{TB}$
110: $16384/f_{TB}$
111: $32768/f_{TB}$

EEPROM Interrupt

The EEPROM Interrupt, is contained within the Multi-function Interrupt. An EEPROM Interrupt request will take place when the EEPROM Interrupt request flag, DEF, is set, which occurs when an EEPROM Write cycle ends. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, EEPROM Interrupt enable bit, DEE, and associated Multi-function interrupt enable bit, must first be set. When the interrupt is enabled, the stack is not full and an EEPROM Write cycle ends, a subroutine call to the respective Multi-function Interrupt vector, will take place. When the EEPROM Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the Multi-function interrupt request flag will be also automatically cleared. As the DEF flag will not be automatically cleared, it has to be cleared by the application program.

LVD Interrupt

The Low Voltage Detector Interrupt is contained within the Multi-function Interrupt. An LVD Interrupt request will take place when the LVD Interrupt request flag, LVF, is set, which occurs when the Low Voltage Detector function detects a low power supply voltage. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, Low Voltage Interrupt enable bit, LVE, and associated Multi-function interrupt enable bit, must first be set. When the interrupt is enabled, the stack is not full and a low voltage condition occurs, a subroutine call to the Multi-function Interrupt vector, will take place. When the Low Voltage Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the Multi-function interrupt request flag will be also automatically cleared. As the LVF flag will not be automatically cleared, it has to be cleared by the application program.

Serial Interface Module Interrupt

The Serial Interface Module Interrupt, also known as the SIM interrupt, is contained within the Multi-function Interrupt. A SIM Interrupt request will take place when the SIM Interrupt request flag, SIMF, is set, which occurs when a byte of data has been received or transmitted by the SIM interface. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and the Serial Interface Interrupt enable bit, SIME, and Multi-function interrupt enable bits, must first be set. When the interrupt is enabled, the stack is not full and a byte of data has been transmitted or received by the SIM interface, a subroutine call to the respective Multi-function Interrupt vector, will take place. When the Serial Interface Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the Multi-function interrupt request flag will be also automatically cleared. As the SIMF flag will not be automatically cleared, it has to be cleared by the application program.

SPIA Interface Interrupt

The SPIA Interface Interrupt is contained within the Multi-function Interrupt. A SPIA Interrupt request will take place when the SPIA Interrupt request flag, SPIAF, is set, which occurs when a byte of data has been received or transmitted by the SPIA interface. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and the SPIA Interface Interrupt enable bit, SPIAE, and Multi-function interrupt enable bits, must first be set. When the interrupt is enabled, the stack is not full and a byte of data has been transmitted or received by the SPIA interface, a subroutine call to the respective Multi-function Interrupt vector, will take place. When the SPIA Interface Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the Multi-function interrupt request flag will be also automatically cleared. As the SPIAF flag will not be automatically cleared, it has to be cleared by the application program.

TM Interrupt

The Compact and Standard Type TMs have two interrupts each while the Enhanced Type TM has three interrupts. All of the TM interrupts are contained within the Multi-function Interrupts. For each of the Compact and Standard Type TMs there are two interrupt request flags TnPF and TnAF and two enable bits TnPE and TnAE. For the Enhanced Type TM there are three interrupt request flags TnPF, TnAF and TnBF and three enable bits TnPE, TnAE and TnBE. A TM interrupt request will take place when any of the TM request flags are set, a situation which occurs when a TM comparator P, A or B match situation happens.

To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, respective TM Interrupt enable bit, and relevant Multi-function Interrupt enable bit, MFnE, must first be set. When the interrupt is enabled, the stack is not full and a TM comparator match situation occurs, a subroutine call to the relevant Multi-function Interrupt vector locations, will take place. When the TM interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the related MFnF flag will be automatically cleared. As the TM interrupt request flags will not be automatically cleared, they have to be cleared by the application program.

Interrupt Wake-up Function

Each of the interrupt functions has the capability of waking up the microcontroller when in the SLEEP or IDLE Mode. A wake-up is generated when an interrupt request flag changes from low to high and is independent of whether the interrupt is enabled or not. Therefore, even though the devices are in the SLEEP or IDLE Mode and its system oscillator stopped, situations such as external edge transitions on the external interrupt pins, a low power supply voltage or comparator output change may cause their respective interrupt flag to be set high and consequently generate an interrupt. Care must therefore be taken if spurious wake-up situations are to be avoided. If an interrupt wake-up function is to be disabled then the corresponding interrupt request flag should be set high before the devices enter the SLEEP or IDLE Mode. The interrupt enable bits have no effect on the interrupt wake-up function.

Programming Considerations

By disabling the relevant interrupt enable bits, a requested interrupt can be prevented from being serviced, however, once an interrupt request flag is set, it will remain in this condition in the interrupt register until the corresponding interrupt is serviced or until the request flag is cleared by the application program.

Where a certain interrupt is contained within a Multi-function interrupt, then when the interrupt service routine is executed, as only the Multi-function interrupt request flags, MFnF, will be automatically cleared, the individual request flag for the function needs to be cleared by the application program.

It is recommended that programs do not use the CALL instruction within the interrupt service subroutine. Interrupts often occur in an unpredictable manner or need to be serviced immediately. If only one stack is left and the interrupt is not well controlled, the original control sequence will be damaged once a CALL subroutine is executed in the interrupt subroutine.

Every interrupt has the capability of waking up the microcontroller when it is in SLEEP or IDLE Mode, the wake up being generated when the interrupt request flag changes from low to high. If it is required to prevent a certain interrupt from waking up the microcontroller then its respective request flag should be first set high before enter SLEEP or IDLE Mode.

As only the Program Counter is pushed onto the stack, then when the interrupt is serviced, if the contents of the accumulator, status register or other registers are altered by the interrupt service program, their contents should be saved to the memory at the beginning of the interrupt service routine.

To return from an interrupt subroutine, either a RET or RETI instruction may be executed. The RETI instruction in addition to executing a return to the main program also automatically sets the EMI bit high to allow further interrupts. The RET instruction however only executes a return to the main program leaving the EMI bit in its present zero state and therefore disabling the execution of further interrupts.

Low Voltage Detector – LVD

Each device has a Low Voltage Detector function, also known as LVD. This enabled the devices to monitor the power supply voltage, V_{DD}, and provide a warning signal should it fall below a certain level. This function may be especially useful in battery applications where the supply voltage will gradually reduce as the battery ages, as it allows an early warning battery low signal to be generated. The Low Voltage Detector also has the capability of generating an interrupt signal.

LVD Register

The Low Voltage Detector function is controlled using a single register with the name LVDC. Three bits in this register, VLVD2~VLVD0, are used to select one of eight fixed voltages below which a low voltage condition will be detemined. A low voltage condition is indicated when the LVDO bit is set. If the LVDO bit is low, this indicates that the V_{DD} voltage is above the preset low voltage value. The LVDEN bit is used to control the overall on/off function of the low voltage detector. Setting the bit high will enable the low voltage detector. Clearing the bit to zero will switch off the internal low voltage detector circuits. As the low voltage detector will consume a certain amount of power, it may be desirable to switch off the circuit when not in use, an important consideration in power sensitive battery powered applications.

LVDC Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	LVDO	LVDEN	—	VLVD2	VLVD1	VLVD0
R/W	—	—	R	R/W	—	R/W	R/W	R/W
POR	—	—	0	0	—	0	0	0

Bit 7~6 Unimplemented, read as “0”

Bit 5 **LVDO:** LVD Output Flag
0: No Low Voltage Detect
1: Low Voltage Detect

Bit 4 **LVDEN:** Low Voltage Detector control
0: Disable
1: Enable

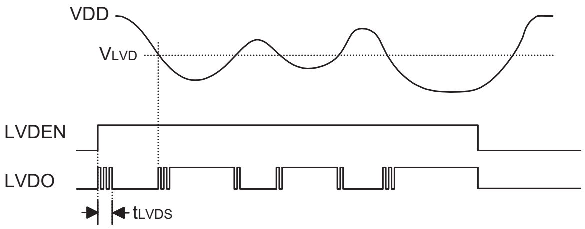
Bit 3 Unimplemented, read as “0”

Bit 2~0 **VLVD2~VLVD0:** Select LVD Voltage
000: 2.0V
001: 2.2V
010: 2.4V
011: 2.7V
100: 3.0V
101: 3.3V
110: 3.6V
111: 4.0V

LVD Operation

The Low Voltage Detector function operates by comparing the power supply voltage, V_{DD}, with a pre-specified voltage level stored in the LVDC register. This has a range of between 2.0V and 4.0V. When the power supply voltage, V_{DD}, falls below this pre-determined value, the LVDO bit will be set high indicating a low power supply voltage condition. The Low Voltage Detector function is supplied by a reference voltage which will be automatically enabled. When the devices are powered down the low voltage detector will remain active if the LVDEN bit is high. After enabling the Low Voltage Detector, a time delay t_{LVDS} should be allowed for the circuitry to stabilise before reading the LVDO bit. Note also that as the V_{DD} voltage may rise and fall rather slowly, at the voltage nears that of V_{LVD}, there may be multiple bit LVDO transitions.

The Low Voltage Detector also has its own interrupt which is contained within one of the Multi-function interrupts, providing an alternative means of low voltage detection, in addition to polling the LVDO bit. The interrupt will only be generated after a delay of t_{LVD} after the LVDO bit has been set high by a low voltage condition. When the devices are powered down the Low Voltage Detector will remain active if the LVDEN bit is high. In this case, the LVF interrupt request flag will be set, causing an interrupt to be generated if V_{DD} falls below the preset LVD voltage. This will cause the devices to wake-up from the SLEEP or IDLE Mode, however if the Low Voltage Detector wake up function is not required then the LVF flag should be first set high before the devices enter the SLEEP or IDLE Mode.



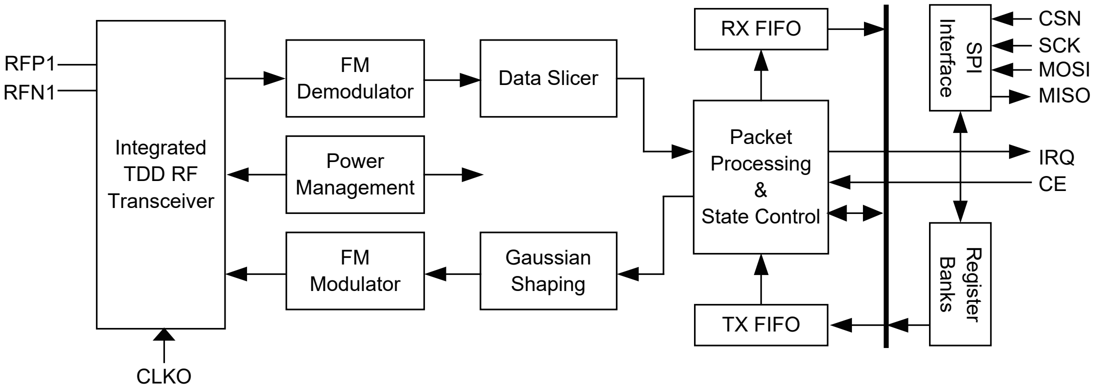
LVD Operation

RF Transceiver

The RF transceiver operates in the world wide ISM frequency band of 2400~2483.5MHz. Burst mode transmission and up to 2Mbps air data rate make them suitable for applications requiring ultra low power consumption. It operates as either a transmitter or a receiver in the Time Division Duplex mode, abbreviated as TDD.

The RF channel frequency determines the center of the channel used by the transceiver. The frequency is set by configuring the RF_CH register in register bank 0 according to the following formula: F0=2400 + RF_CH (MHz). The resolution of the RF channel frequency is 1MHz.

A transmitter and a receiver must be programmed with the same RF channel frequency to be able to communicate with each other. The output power of the transceiver is set by the RF_PWR bits in the RF_SETUP register. Demodulation is implemented with the embedded data slicer and bit recovery logic. The air data rate can be programmed to be 250Kbps, 1Mbps or 2Mbps by configuring the RF_DR_HIGH and RF_DR_LOW registers. A transmitter and a receiver must be programmed with the same setting.



RF Transceiver Block Diagram

RF Transceiver Abbreviations

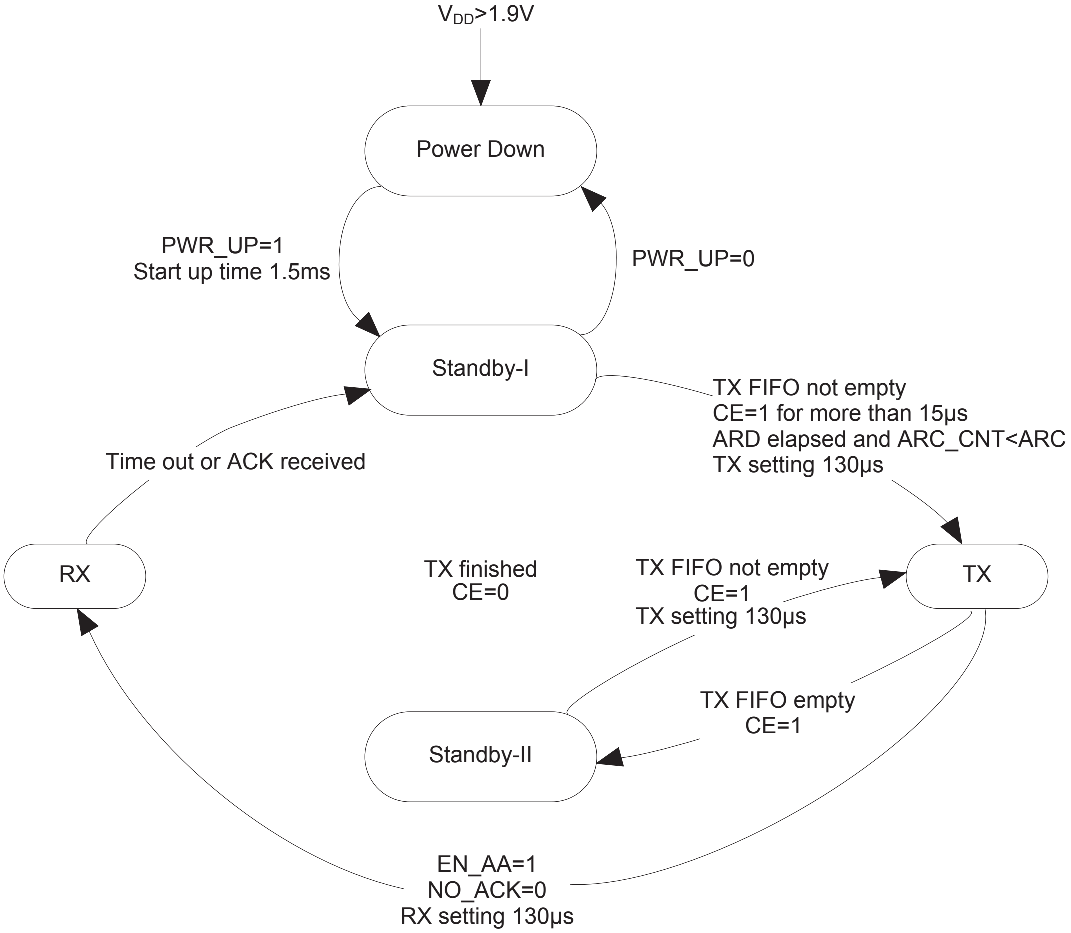
ACK	Acknowledgement
ARC	Auto Retransmission Count
ARD	Auto Retransmission Delay
CD	Carrier Detection
CE	Chip Enable
CRC	Cyclic Redundancy Check
CSN	Chip Select Not
DPL	Dynamic Payload Length
FIFO	First-In-First-Out
GFSK	Gaussian Frequency Shift Keying
GHz	Gigahertz
LNA	Low Noise Amplifier
IRQ	Interrupt Request
ISM	Industrial-Scientific-Medical
LSB	Least Significant Bit
MAX_RT	Maximum Retransmit
Mbps	Megabit per second
MCU	Microcontroller Unit
MHz	Megahertz
MISO	Master In Slave Out
MOSI	Master Out Slave In
MSB	Most Significant Bit
PA	Power Amplifier
PID	Packet Identity Bits
PLD	Payload
PRX	Primary RX
PTX	Primary TX
PWD_DWN	Power Down
PWD_UP	Power Up
RF_CH	Radio Frequency Channel
RSSI	Received Signal Strength Indicator
RX	Receive
RX_DR	Receive Data Ready
SCK	SPI Clock
SPI	Serial Peripheral Interface
TDD	Time Division Duplex
TX	Transmit
TX_DS	Transmit Data Sent
XTAL	Crystal

RF Transceiver State Control

The RF transceiver block includes an integrated a state machine that controls the state transition between different modes. When the auto acknowledge feature is disabled, the state transition will be fully controlled by the MCU.

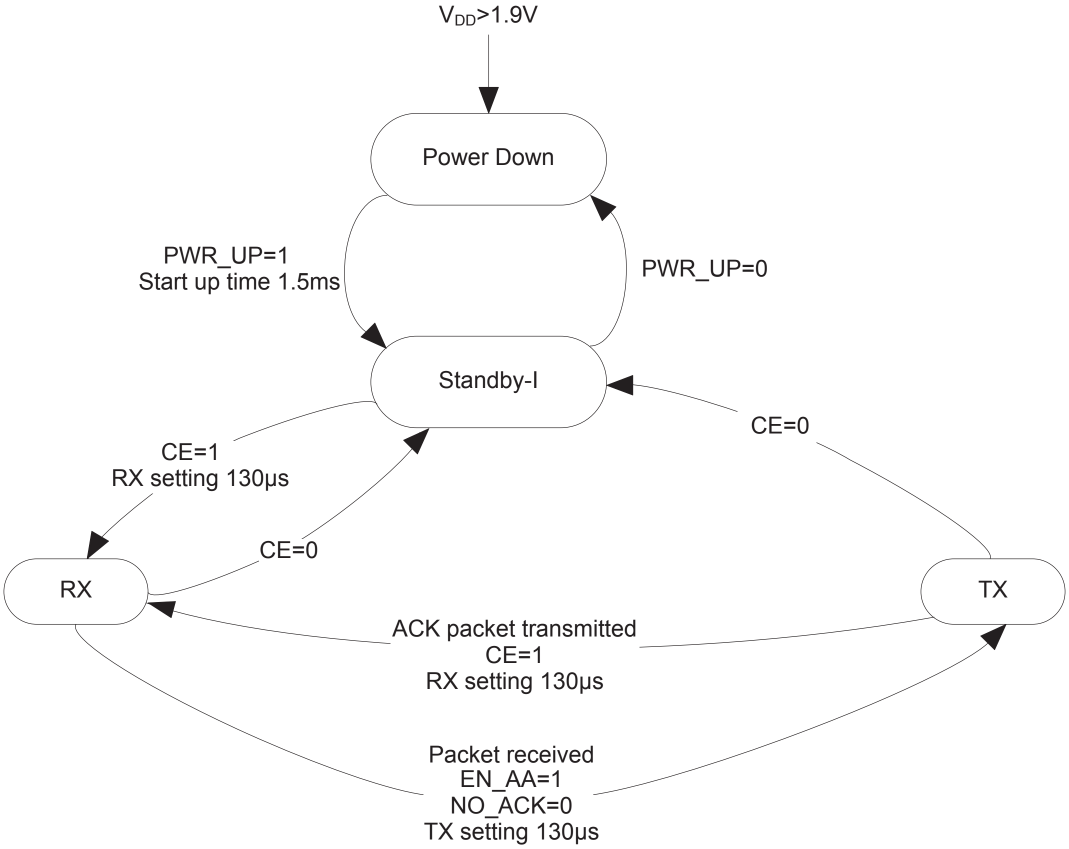
- SPI register: PWR_UP, PRIM_RX, EN_AA, NO_ACK, ARC, ARD
- System information: Time out, ACK received, ARD elapsed, ARC_CNT, TX FIFO empty, ACK packet transmitted, Packet received

- **Primary Transmission State Control – PTX, PRIM_RX=0**



Primary Transmission State Diagram

- **Primary Reception State Control – PRX, PRIM_RX=1**



Primary Reception State Diagram

Power Down Mode

The Power down mode of the RF Transceiver is entered by setting the PWR_UP bit in the CONFIG register to low. In the power down mode the transceiver block is in the sleep mode where it has minimal current consumption. However, the SPI interface is still active in this mode and all register values can be configured by the SPI interface.

The MCU and RF Transceiver are powered down independently of each other. The method of powering down the MCU is covered in the previous MCU section of the datasheet. The RF Transceiver must be powered down before the MCU is powered down. This is implemented by first clearing the PWR_UP bit in the CONFIG register to disable the RF Transceiver circuitry and then pulling the $\overline{\text{SCSA}}$ line to high to disable the RF SPI interface circuitry. After the RF transceiver is completely powered down, the MCU can be powered down to minimise the power consumption.

Standby-I Mode

By setting the PWR_UP bit in the CONFIG register to 1 and de-asserting the CE signal to a low state, the devices enter the Standby-I mode. The Standby-I mode is used to minimise the average current consumption and also to maintain a shorter start-up time. In this mode, the clock is still active. When the CE signal is set to a low state, the transceiver will return to the Standby-I mode regardless of whether it is in the TX or RX mode.

Standby-II Mode

In the Standby-II mode more RF Transceiver circuitry is active than in the Standby-I mode and therefore more current is consumed. The transceiver will enter the Standby-II mode from the TX mode when the CE signal is set to a high state and the TX FIFO is empty. If a new data packet is uploaded into the TX FIFO, the devices will automatically return to the TX mode and the packet will be transmitted.

TX Mode

- **Primary Transmit Device – PTX, PRIM_RX=0**

The TX mode is an active mode where the PTX device transmits a data packet. To enter this mode from power down mode, the PTX device must set the PWR_UP bit to high, the PRIM_RX bit to low, a payload in the TX FIFO and generate a high pulse on the CE line for more than 15μs.

The PTX device stays in the TX mode until it finishes transmitting the current packet and then enters the Standby-II mode. If the CE signal is in a low state, it will return to Standby-I mode. If the CE signal is in a high state, the next action will be determined by the status of the TX FIFO. If the TX FIFO is not empty, the PTX device will enter the TX mode and then transmit the next packet. If the TX FIFO is empty, the PTX device will remain in the Standby-II mode. It is important to note that it never remains in the TX mode for more than 4ms during a transmit operation.

If the auto retransmit is enabled by setting the EN_AA bit to 1 and an auto acknowledge is required by setting the NO_ACK bit to 0, the PTX device will enter the TX mode from the Standby-I mode when the ARD has elapsed and the number of retry is less than the ARC.

The PTX device will enter the RX mode from the TX mode only when the EN_AA bit is set to 1 and the NO_ACK bit is cleared to 0 to receive the acknowledge packet.

RX Mode

- **Primary Receive Device – PRX, PRIM_RX=1**

The RX mode is an active mode where the transceiver is configured as a receiver. To enter this RX mode from the Standby-I mode, the PRX device must set the PWR_UP bit to high, the PRIM_RX bit to high and the CE signal to high. In this mode the receiver demodulates the signals from the RF channel, constantly presenting the demodulated data to the packet processing engine. The packet processing engine continuously searches for a valid packet. If a valid packet is found by a matching address and a valid CRC, the packet payload is presented to a vacant slot in the RX FIFO. If the RX FIFO is full, the received packet will be discarded.

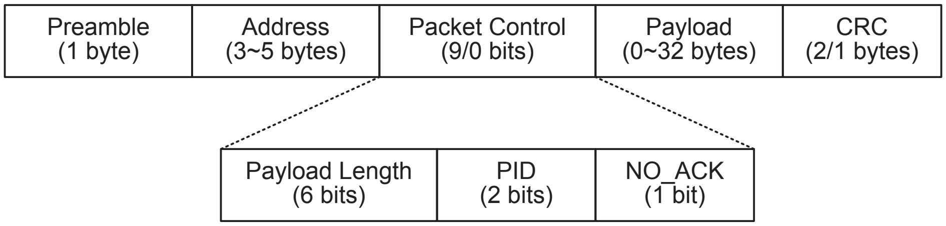
The PRX device remains in the RX mode until the MCU configures it to enter the Standby-I mode or Power Down mode. In the RX mode a carrier detection signal, CD, is made available. The CD signal is set high when an RF signal is detected on the receiving frequency channel. The internal CD signal is filtered before being written into the CD register. The RF signal must be present for at least 128μs before the CD signal is set high.

The PRX device will enter the TX mode from the RX mode only when the EN_AA bit is set to 1 and the NO_ACK bit is cleared to 0 in the received packet to transmit an acknowledge packet with a pending payload in the TX FIFO.

RF Transceiver Packet Processing

Packet Format

The complete packet includes a preamble, 3~5 address bytes, a packet control field, 0~32 payload bytes and a CRC field.



Packet Format

Preamble

The preamble is a bit sequence used to detect the “0” and “1” levels in the receiver. The preamble is one byte long whose value is either 01010101 or 10101010. If the first bit in the address is 1, the preamble is automatically set to 10101010 while if the first bit is 0, the preamble is automatically set to 01010101. This is done to ensure there are enough transitions in the preamble to stabilise the receiver.

Address

This field is the address for the receiver. An address ensures that the packet is detected by the target receiver. The address field can be configured to be 3, 4 or 5 bytes long by the AW register. The PRX device can open up to six data pipes to support up to six PTX devices with specific addresses. All six PTX device addresses are searched simultaneously. In the PRX device, the data pipes are enabled with the corresponding control bits in the EN_RXADDR register. The default status is that only data pipe 0 and pipe 1 are enabled. Each data pipe address can be configured in the RX_ADDR_PX registers. Each pipe can have up to a 5 byte configurable address. Data pipe 0 has a unique 5-byte address. Data pipes 1~5 share the 4 most significant address bytes. The least significant byte must be unique for all 6 pipes.

To ensure that the ACK packet from the PRX is transmitted to the correct PTX, the PRX takes the data pipe address where it received the packet and uses it as the TX address when transmitting the ACK packet.

On the PRX device the RX_ADDR_Pn defined as the pipe address must be unique. On the PTX device the TX_ADDR must be the same as the RX_ADDR_P0 on the PTX, and as the pipe address for the designated pipe on the PRX. No other data pipe can receive data until a complete packet is received by a data pipe that has detected its address. When multiple PTX devices are transmitting to a PRX, the ARD can be used to skew the auto retransmission so that they only block each other once.

Packet Control

When the Dynamic Payload Length function is enabled, the packet control field contains a 6-bit payload length field, a 2-bit Packet Identity, PID, field and a 1-bit NO_ACK flag.

- **Payload Length**

  The payload length field is used to define the payload length and only used if the Dynamic Payload Length function is enabled.

- **PID**

  The 2-bit PID field is used to detect whether the received packet is new or retransmitted. The PID prevents the PRX device from presenting the same payload more than once to the MCU. The PID field is incremented at the TX side for each new packet received through the SPI interface. The PID and CRC fields are used by the PRX device to determine whether a packet is old or new. When several data packets are lost on the link, the PID fields may become equal to the last received PID. If a packet has the same PID as the previous packet, the transceiver compares the CRC sums from both packets. If the CRC sums are also the same, the last received packet will be considered a copy of the previously received packet and be discarded.

- **NO_ACK**

  The NO_ACK flag is used only when the auto acknowledgement feature is used. Setting the flag high informs the receiver that the packet is not to be auto acknowledged. The PTX device can set the NO_ACK flag bit in the Packet Control Field with the command: W_TX_PAYLOAD_NOACK. However, the function must first be enabled in the FEATURE register by setting the EN_DYN_ACK bit. When the auto acknowledgement function is used, the PTX will directly enter to the Standby-I mode after the packet is transmitted and the PRX device does not transmit an ACK packet when it receives the PTX transmitted packet.

Payload

The payload field is used to define the contents of the packet. It can be from 0 to 32 bytes wide and transmitted on-air as it is uploaded (unmodified) to the device. The transceiver provides two alternatives for handling payload lengths, static and dynamic payload length. The static payload length of the six data pipes can be individually set.

The default alternative is static payload length. With static payload length all packets between a transmitter and a receiver have the same length. The static payload length is set by the RX_PW_Pn registers. The payload length on the transmitter side is set by the number of bytes clocked into the TX_FIFO and must equal to the value in the RX_PW_Pn register on the receiver side. Each pipe has its own payload length.

The Dynamic Payload Length, DPL, is an alternative to the static payload length. The DPL enables the transmitter to send packets with variable payload lengths to the receiver. This means that for a system with different payload lengths it is not necessary to scale the packet length to the longest payload.

With the DPL feature the transceiver can decode the payload length of the received packet automatically instead of using the RX_PW_Pn registers. The MCU can read the length of the received payload using the command: R_RX_PL_WID. In order to enable the DPL function, the EN_DPL bit in the FEATURE register must be set. In the RX mode the DYNPD register has to be properly configured. A PTX device must set the DPL_P0 bit in the DYNPD register to transmit to a PRX with the DPL function being enabled.

CRC

The CRC is the error detection mechanism in the packet. The number of bytes in the CRC is set by the CRCO bit in the CONFIG register. It may be either 1 or 2 bytes and is calculated over the address, Packet Control Field and Payload. The polynomial for a1-byte CRC is $X^8 + X^2 + X + 1$ with an initial value of 0xFFH. The polynomial for a 2-byte CRC is $X^{16} + X^{12} + X^5 + 1$ with an initial value of 0xFFFFH. No packet is accepted by the receiver side if the CRC fails.

Packet Handling

The transceiver circuitry uses burst mode for payload transmission and reception.

The transmitter fetches the payload from the TX FIFO, automatically assembles it into a packet and transmits the packet in a very short burst period with a 1Mbps or 2Mbps air data rate. After a transmission, if the PTX packet has the NO_ACK flag set, the transceiver sets the TX_DS bit and gives an active low interrupt pulse on the IRQ line sent to the MCU. If the PTX is an ACK packet, the PTX needs to receive an ACK from the PRX and then assert the TX_DS IRQ.

The receiver automatically validates and disassembles the received packet. If there is a valid packet within the new payload, it will write the payload into the RX FIFO, set the RX_DR bit and give an active low interrupt pulse on the IRQ line sent to the MCU.

When the auto acknowledge function is enabled by setting the EN_AA bit to 1, the PTX device will automatically wait for an acknowledge packet after transmission and re-transmit the original packet after the ARD delay until an acknowledge packet is received or the number of re-transmission exceeds a threshold defined by the ARC field. If the number of re-transmissions exceeds a threshold defined by the ARC field, the transceiver will set the MAX_RT bit and give an active low interrupt pulse on the IRQ line sent to the MCU. Two packet loss counters, ARC_CNT and PLOS_CNT, are incremented by one each time a packet is lost. The ARC_CNT counter counts the number of retransmissions for the current transaction. The PLOS_CNT counter counts the total number of retransmissions since the last channel change. The ARC_CNT counter is reset by initiating a new transaction while the PLOS_CNT counter is reset by writing a value to the RF_CH register to change a RF channel. It is possible to use the information in the OBSERVE_TX register to make an overall assessment of the channel quality.

The PTX device will retransmit if its RX FIFO is full but the received ACK frame has a payload. As an alternative for the PTX device to auto retransmit it is possible to manually set the transceiver to retransmit a packet a number of times. This is done using the REUSE_TX_PL command. When the auto acknowledge function is enabled, the PRX device will automatically check the NO_ACK field in the received packet, and if the NO_ACK bit is 0, it will automatically send an acknowledge packet to the PTX device. If the EN_ACK_PAY bit is set, the acknowledge packet can also be regarded as a pending payload in the TX FIFO.

RF Transceiver Data and Control Interface

TX/RX FIFO

The data FIFOs are used to store the payload that is to be transmitted in the TX FIFO or is received in the RX FIFO and ready to be clocked out. The FIFO is accessible in both the PTX mode and PRX modes. There is a three level 32 byte FIFO for both the TX and RX, supporting both the acknowledge mode or no acknowledge mode with up to six data pipes.

- TX three levels, 32 bytes FIFO
- RX three levels, 32 bytes FIFO

Both the TX and RX FIFOs have a controller and are accessible through the SPI interface using dedicated SPI commands. A TX FIFO in PRX can store the payload for ACK packets for three different PTX devices. If the TX FIFO contains more than one payload for a pipe, the different payloads are handled using the first-in first-out principle. The TX FIFO in a PRX device is blocked if all pending payloads are addressed to pipes where the link to the PTX device is lost. In this case, the MCU can flush the TX FIFO using the FLUSH_TX command.

The RX FIFO in the PRX device may contain a payload from up to three different PTX devices. A TX FIFO in the PTX devices can have up to three payloads stored. The TX FIFO can be written to by three commands, W_TX_PAYLOAD and W_TX_PAYLOAD_NO_ACK in the PTX mode and W_ACK_PAYLOAD in the PRX mode. All three commands give access to the TX_PLD register. The RX FIFO can be read by the command R_RX_PAYLOAD in both PTX and PRX modes. This command gives access to the RX_PLD register. The payload in the TX FIFO in a PTX device is NOT removed if the MAX_RT IRQ is asserted.

In the FIFO_STATUS register it is possible to know whether the TX and RX FIFO are full or empty. The TX_REUSE bit is also available in the FIFO_STATUS register. The TX_REUSE bit is set by the SPI command, REUSE_TX_PL, and is reset by the SPI command, W_TX_PAYLOAD or FLUSH TX.

Interrupt

In the RF transceiver circuitry there is an active low interrupt line, IRQ, which is activated when the TX_DS IRQ, RX_DR IRQ or MAX_RT IRQ bit is set to high by the state machine in the STATUS register. The IRQ line is internally connected to the MCU external interrupt input. The detailed MCU external interrupt configurations are described in the preceding section in this datasheet. The IRQ line is reset when the MCU writes a ‘1’ into the IRQ source bit in the STATUS register. The IRQ mask control bit in the CONFIG register is used to select which IRQ source is allowed to assert the IRQ line. By setting one of the MASK bits high, the corresponding IRQ source will be disabled. By default all IRQ sources are enabled. The 3-bit pipe information in the STATUS register is updated when the IRQ line changes state from high to low. If the STATUS register is read during a high to low transition of an IRQ line, the pipe information is not available.

SPI Interface

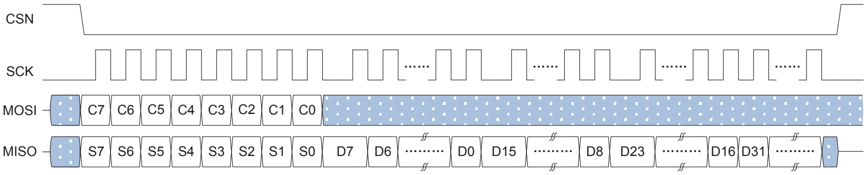
The SPI commands are shown in the following table. Every new command must be started by a high to low transition on the CSN line. In parallel to the SPI command byte applied on the MOSI line, the STATUS register bit, Sn, is serially shifted out on the MISO line. The RF Transceiver SPI interface is internally connected to the MCU SPIA interface. The detailed MCU SPIA interface operations are described in the corresponding SPIA interface section in this datasheet.

The serial shifting SPI commands are in the following format:

- Command byte: one byte command word from MSB to LSB
- Data read operation: The data byte is shifted out from the least significant byte to the most significant byte and MSB first in each byte for all registers in all register banks.
- Data write operation: The data byte is shifted in from the most significant byte to the least significant byte and MSB in each byte first for register 0 to register 8 in register bank 1. For other registers in all banks the data byte is shifted in from the least significant byte to the most significant byte.

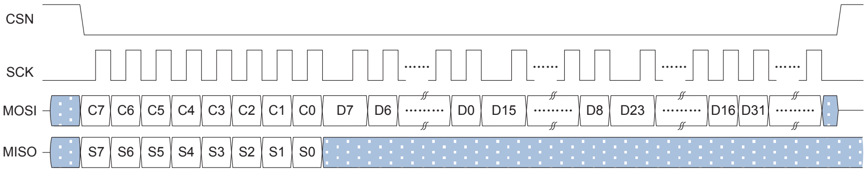
Command name	Command byte (Cn)	Data bytes (Dn)	Operation
R_REGISTER	000A AAAA	1 to 5	Read command and status registers AAAAA=5-bit Register Map Address
W_REGISTER	001A AAAA	1 to 5	Write command and status registers AAAAA=5-bit Register Map Address Executable in power down or standby modes only
R_RX_PAYLOAD	0110 0001	1 to 32 LSB byte first	Read RX-payload: 1~32 bytes, used in the RX mode A read operation always starts at byte 0 The RX Payload is implemented in register bank 0 and is deleted from the FIFO after it is read.
W_TX_PAYLOAD	1010 0000	1 to 32 LSB byte first	Write TX-payload: 1~32 bytes A write operation always starts at byte 0 The TX Payload is implemented in register bank 0 and is used in the TX mode
FLUSH_TX	1110 0001	0	Flush TX FIFO, used in TX mode
FLUSH_RX	1110 0010	0	Flush RX FIFO, used in RX mode. Should not be executed during transmission of acknowledge, that is, an acknowledge package will not be completed.
REUSE_TX_PL	1110 0011	0	Used for a PTX device Reuse last transmitted payload Packets are repeatedly retransmitted as long as CE is high. The TX payload reuse is active until W_TX_PAYLOAD or FLUSH TX is executed. The TX payload reuse must not be activated or deactivated during package transmission.

Command name	Command byte (Cn)	Data bytes (Dn)	Operation
ACTIVATE	0101 0000	1	This write command followed by data 0x73 activates the following features: • R_RX_PL_WID • W_ACK_PAYLOAD • W_TX_PAYLOAD_NOACK A new ACTIVATE command with the same data deactivates them again. This is executable in power down or standby modes only. The features registers named as R_RX_PL_WID, W_ACK_PAYLOAD, and W_TX_PAYLOAD_NOACK are initially in a deactivated state; a write operation has no effect, a read operation only results in zeros on MISO. To activate these registers, use the ACTIVATE command followed by data 0x73. Then they can be accessed just like any other registers. Using the same command and data will deactivate the registers again. This write command followed by data 0x53 toggles the register bank and the current register bank number can be read out from the STATUS register bit 7.
R_RX_PL_WID	0110 0000		Read RX-payload width for the top R_RX_PAYLOAD in the RX FIFO
W_ACK_PAYLOAD	1010 1PPP	1 to 32 LSB byte first	Write ACK Payload: 1~32 bytes, used in the RX mode Write Payload to be transmitted together with ACK packet on PIPE "PPP" where the "PPP" ranges from 000 to 101. Maximum three ACK packet payloads can be pending. Payloads with the same PPP are handled using first-in- first-out principle. A write operation to the ACK payload implemented in the register bank 0 always starts at byte 0.
W_TX_PAYLOAD_NOACK	1011 0000	1 to 32 LSB byte first	Used in TX mode and implemented in the register bank 0 Disables AUTOACK on this specific packet.
NOP	1111 1111	0	No Operation. Might be used to read the STATUS register.



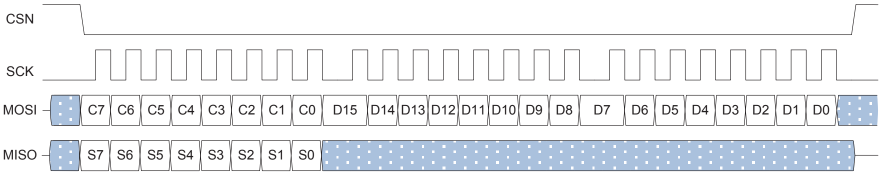
Note: The data byte on the MISO line may be 1 byte, 4 bytes, 11 bytes or 32 bytes depending upon which register is accessed.

SPI read operation – All registers in Bank 0 & Bank1



Note: The data byte on the MISO line may be 1 byte, 4 bytes, 11 bytes or 32 bytes depending upon which register is accessed.

SPI write operation – All registers in Bank 0 & Register 9~14 in Bank1


CSN
SCK
MOSI
C7 C6 C5 C4 C3 C2 C1 C0 D15 D14 D13 D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0
MISO
S7 S6 S5 S4 S3 S2 S1 S0

SPI write operation – Register 0~8 in Bank1

RF Transceiver Register Map

There are two register banks, which can be toggled by an SPI "ACTIVATE" command followed with 0x53 data byte and the bank status can be read from the STATUS register bit 7 in the Register Bank0.

Register Bank 0

It is recommended that no access is executed on reserved or non defined registers. Otherwise, this may result in unpredictable conditions.

- **Address 00H – CONFIG Register**

This register is used to configure the primary setting of the RF Transceiver.

Bit	7	6	5	4	3	2	1	0
Name	—	MASK_RX_DR	MASK_TX_DS	MASK_MAX_RT	EN_CRC	CRCO	PWR_UP	PRIM_RX
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	1	0	0	0

Bit 7 Reserved bit, only data "0" allowed.

Bit 6 **MASK_RX_DR:** RX_DR Mask interrupt control
0: RX_DR interrupt is reflected as an active low interrupt pulse on the IRQ line
1: RX_DR interrupt is not reflected on the IRQ line

Bit 5 **MASK_TX_DS:** TX_DS Mask interrupt control
0: TX_DS interrupt is reflected as an active low interrupt pulse on the IRQ line
1: TX_DS interrupt is not reflected on the IRQ line

Bit 4 **MASK_MAX_RT:** MAX_RT Mask interrupt control
0: MAX_RT interrupt is reflected as an active low interrupt pulse on the IRQ line
1: MAX_RT interrupt is not reflected on the IRQ line

Bit 3 **EN_CRC:** CRC function enable control
0: Disable
1: Enable
This bit will be forced to high if one of the bits in the EN_AA register is high.

Bit 2 **CRCO:** CRC encoding scheme
0: 1 byte
1: 2 bytes

Bit 1 **PWR_UP:** RF Transceiver power control
0: Power down
1: Power up

Bit 0 **PRIM_RX:** RX/TX mode selection
0: Primary Transmit – PTX
1: Primary Receive – PRX

- **Address 01H – EN_AA Register**

This register is used to enable the “Auto Acknowledgement” function of the individual data pipe.

Bit	7	6	5	4	3	2	1	0
Name	—	—	ENAA_P5	ENAA_P4	ENAA_P3	ENAA_P2	ENAA_P1	ENAA_P0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	1	1	1	1	1	1

Bit 7~6 Reserved bits, only data “00” allowed.

Bit 5 **ENAA_P5:** Data Pipe 5 Auto Acknowledgement function enable control
0: Disable
1: Enable

Bit 4 **ENAA_P4:** Data Pipe 4 Auto Acknowledgement function enable control
0: Disable
1: Enable

Bit 3 **ENAA_P3:** Data Pipe 3 Auto Acknowledgement function enable control
0: Disable
1: Enable

Bit 2 **ENAA_P2:** Data Pipe 2 Auto Acknowledgement function enable control
0: Disable
1: Enable

Bit 1 **ENAA_P1:** Data Pipe 1 Auto Acknowledgement function enable control
0: Disable
1: Enable

Bit 0 **ENAA_P0:** Data Pipe 0 Auto Acknowledgement function enable control
0: Disable
1: Enable

- **Address 02H – EN_RXADDR Register**

This register is used to enable the RX data pipe address.

Bit	7	6	5	4	3	2	1	0
Name	—	—	ERX_P5	ERX_P4	ERX_P3	ERX_P2	ERX_P1	ERX_P0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	1	1

Bit 7~6 Reserved bits, only data “00” allowed.

Bit 5 **ERX_P5:** Data Pipe 5 enable control
0: Disable
1: Enable

Bit 4 **ERX_P4:** Data Pipe 4 enable control
0: Disable
1: Enable

Bit 3 **ERX_P3:** Data Pipe 3 enable control
0: Disable
1: Enable

Bit 2 **ERX_P2:** Data Pipe 2 enable control
0: Disable
1: Enable

Bit 1 **ERX_P1:** Data Pipe 1 enable control
0: Disable
1: Enable

Bit 0 **ERX_P0:** Data Pipe 0 enable control
0: Disable
1: Enable

- **Address 03H – SETUP_AW Register**

This register is used to specify the data pipe address field width.

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	AW1	AW0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	1	1

Bit 7~2 Reserved bits, only data “000000” allowed.

Bit 1~0 **AW1~AW0:** Data pipe Address Field Width
00: Illegal, can not be used.
01: 3 bytes
10: 4 bytes
11: 5 bytes
The field is used to specify the data pipe address field width which is the same for all data pipes.

- **Address 04H – SETUP_RETR Register**

This register is used to set the number and delay of the automatic retransmission.

Bit	7	6	5	4	3	2	1	0
Name	ARD3	ARD2	ARD1	ARD0	ARC3	ARC2	ARC1	ARC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	1	1

Bit 7~4 **ARD3~ARD0:** Automatic Retransmission Delay selection
0000: (1×250) μs
0001: (2×250) μs
0010: (3×250) μs
:
1110: (15×250) μs
1111: (16×250) μs

The Automatic Retransmission Delay is defined as the time from the end of the current transmission to the start of the next transmission.

Bit 3~0 **ARC3~ARC0:** Automatic Retransmission Number
0000: No retransmission when automatic acknowledgement failed.
0001: 1 retransmission when automatic acknowledgement failed.
0010: 2 retransmissions when automatic acknowledgement failed.
:
1110: 14 retransmissions when automatic acknowledgement failed.
1111: 15 retransmissions when automatic acknowledgement failed.

- **Address 05H – RF_CH Register**

This register is used to select the RF channel.

Bit	7	6	5	4	3	2	1	0
Name	—	RF_CH6	RF_CH5	RF_CH4	RF_CH3	RF_CH2	RF_CH1	RF_CH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	1	0

Bit 7 Reserved bits, only data “0” allowed.

Bit 6~0 **RF_CH6~RF_CH0:** RF Channel selection
0000000~1111111: RF channel 0~RF channel 127

- **Address 06H – RF_SETUP Register**

This register is used to configure the RF output power and data rate.

Bit	7	6	5	4	3	2	1	0
Name	—	—	RF_DR_LOW	PLL_LOCK	RF_DR_HIGH	RF_PWR1	RF_PWR0	LNA_HCURR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	1	1	1	1

Bit 7~6 Reserved bits, only data “00” allowed.

Bit 5 **RF_DR_LOW:** Air Data Rate select bit
This bit is used to select the air data rate together with the RF_DR_HIGH bit. Refer to RF_DR_HIGH bit definition for more details.

Bit 4 **PLL_LOCK:** PLL Signal Lock control
0: No effect
1: PLL signal is locked
This bit is used to lock the PLL signal and only available in the test mode.

Bit 3 **RF_DR_HIGH:** Air Data Rate select bit
[RF_DR_LOW, RF_DR_HIGH]=Air Data Rate
00: 1Mbps
01: 2Mbps
10: 250Kbps
11: 2Mbps

Bit 2~1 **RF_PWR1~RF_PWR0:** RF TX Output Power selection
00: -26dBm
01: -14dBm
10: -6dBm
11: -1dBm
Note: The Optimisation Register 4 is located at address 04H in the RF Bank 1.

Bit 0 **LNA_HCURR:** LNA Gain selection
0: Low gain
1: High gain

- **Address 07H – STATUS Register**

This register is used to store the status during the RF data transfer. When the SPI command is committed to the RF Transceiver on the MOSI line, the content of the STATUS register will be serially shifted out from the RF Transceiver on the MISO line.

Bit	7	6	5	4	3	2	1	0
Name	RBANK	RX_DR	TX_DS	MAX_RT	RX_P_NO2	RX_P_NO1	RX_P_NO0	TX_FULL
R/W	R	R/W	R/W	R/W	R	R	R	R
POR	0	0	0	0	1	1	1	0

Bit 7 **RBANK:** Register Bank Status
0: Register bank 0
1: Register bank 1

Bit 6 **RX_DR:** RX FIFO Data Received Status
0: No data is received in RX FIFO
1: New data has been received in RX FIFO

This bit is set to 1 by hardware and cleared to 0 by writing a "1" into it. When a new data packet is received in the RX FIFO, this bit will be asserted.

Bit 5 **TX_DS:** TX FIFO Data Sent Status
0: No data is transmitted from TX FIFO
1: New data has been transmitted from TX FIFO

This bit is set to 1 by hardware and cleared to 0 by writing a "1" into it. When a new data packet is transmitted from the TX FIFO, this bit will be asserted. If the AUTO_ACK function is enabled, this bit will be set to 1 after the ACK is received.

Bit 4 **MAX_RT:** Maximum TX retransmission Status
0: TX re-transmission number does not reach to the maximum retransmission number.
1: TX re-transmission number has reached to the maximum retransmission number.

This bit is set to 1 by hardware and cleared to 0 by writing a "1" into it. When the TX re-transmission number has reached to the maximum retransmission number, this bit will be asserted. If the MAX_RT bit is set to 1, it must be cleared to 0 by application program to enable further data communication.

Bit 3~1 **RX_P_NO2~RX_P_NO0:** Data pipe number in RX FIFO
000~101: Data pipe 0~Data pipe 5.
110: Not used
111: RX FIFO empty

This field is used to indicate the data pipe number for the available payload in the RX FIFO. When this field is "111", it means that the RX FIFO is empty.

Bit 0 **TX_FULL:** TX FIFO full flag
0: TX FIFO is not full
1: TX FIFO is full

This bit is set and cleared by hardware. When the TX FIFO is full, this bit will be asserted. If this bit is 0, it means that the TX FIFO is not full yet and still has available locations to be used.

- **Address 08H – OBSERVE_TX Register**

This register includes two read-only counters and is used to indicate the status during the TX data transmission.

Bit	7	6	5	4	3	2	1	0
Name	PLOS_CNT3	PLOS_CNT2	PLOS_CNT1	PLOS_CNT0	ARC_CNT3	ARC_CNT2	ARC_CNT1	ARC_CNT0
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~4 **PLOS_CNT3~PLOS_CNT0:** TX lost data packet number
0000~1111: 0 TX lost data packet~15 TX lost data packets
This field is used to count the total lost data packets during the TX data transmission. The PLOS_CNT value will be incremented by one each time a TX data packet is lost and retransmitted. When the counter value is equal to “1111”, it will stop counting and remain the maximum value of 1111 instead of counter overflow until a reset condition occurs. This counter will be reset by a write operation to the RF_CH register.

Bit 3~0 **ARC_CNT3~ARC_CNT0:** Automatic Retransmission number
This field is used to count the retransmitted data packets during the TX data transmission. The ARC_CNT value will be incremented by one each time a TX data packet is lost and retransmitted. This counter will be reset when a new TX data starts to be transmitted.

- **Address 09H – CD Register**

This register is used to indicate the carrier detect status.

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	—	CD
R/W	—	—	—	—	—	—	—	R
POR	—	—	—	—	—	—	—	0

Bit 7~1 Unimplemented, read as “0”

Bit 0 **CD:** Carrier Detect status
0: No carrier is detected
1: Carrier has been detected

- **Address 0AH – RX_ADDR_P0 Register**

This register is used to specify the data pipe 0 RX address.

Byte	4	3	2	1	0
Name	RX_ADDR_P0 [39:32]	RX_ADDR_P0 [31:24]	RX_ADDR_P0 [23:16]	RX_ADDR_P0 [15:8]	RX_ADDR_P0 [7:0]
R/W	R/W	R/W	R/W	R/W	R/W
POR	E7	E7	E7	E7	E7

Bit 39~0 **RX_ADDR_P0:** Data pipe0 receive address
This field is used to define the data pipe 0 receive address. The address field width can be up to 5 bytes which is specified in the SETUP_AW register. This address field configuration is carried out in a specific way from the least significant byte to the most significant byte.

- **Address 0BH – RX_ADDR_P1 Register**

This register is used to specify the data pipe 1 RX address.

Byte	4	3	2	1	0
Name	RX_ADDR_P1 [39:32]	RX_ADDR_P1 [31:24]	RX_ADDR_P1 [23:16]	RX_ADDR_P1 [15:8]	RX_ADDR_P1 [7:0]
R/W	R/W	R/W	R/W	R/W	R/W
POR	C2	C2	C2	C2	C2

Bit 39~0 **RX_ADDR_P1:** Data pipe1 receive address

This field is used to define the data pipe 1 receive address. The address field width can be up to 5 bytes which is specified in the SETUP_AW register. This address field configuration is carried out in a specific way from the least significant byte to the most significant byte.

- **Address 0CH – RX_ADDR_P2 Register**

This register is used to specify the data pipe 2 RX address.

Byte	4	3	2	1	0
Name	RX_ADDR_P1 [39:32]	RX_ADDR_P1 [31:24]	RX_ADDR_P1 [23:16]	RX_ADDR_P1 [15:8]	RX_ADDR_P2 [7:0]
R/W	R/W	R/W	R/W	R/W	R/W
POR	C2	C2	C2	C2	C3

Bit 39~8 **RX_ADDR_P1 [39:8]:** Data pipe1 receive address bit 39~bit 8

Bit 7~0 **RX_ADDR_P2 [7:0]:** Data pipe 2 receive address bit 7~bit 0

This field is used to define the data pipe 2 receive address. Note that only bit 7~bit 0 of the address field can be configured and other 32MSBs from bit 39 to bit 8 must be the same as the data pipe 1 address.

- **Address 0DH – RX_ADDR_P3 Register**

This register is used to specify the data pipe 3 RX address.

Byte	4	3	2	1	0
Name	RX_ADDR_P1 [39:32]	RX_ADDR_P1 [31:24]	RX_ADDR_P1 [23:16]	RX_ADDR_P1 [15:8]	RX_ADDR_P3 [7:0]
R/W	R/W	R/W	R/W	R/W	R/W
POR	C2	C2	C2	C2	C4

Bit 39~8 **RX_ADDR_P1 [39:8]:** Data pipe1 receive address bit 39~bit 8

Bit 7~0 **RX_ADDR_P3 [7:0]:** Data pipe 3 receive address bit 7~bit 0

This field is used to define the data pipe 3 receive address. Note that only bit 7~bit 0 of the address field can be configured and other 32MSBs from bit 39 to bit 8 must be the same as the data pipe 1 address.

- **Address 0EH – RX_ADDR_P4 Register**

This register is used to specify the data pipe 4 RX address.

Byte	4	3	2	1	0
Name	RX_ADDR_P1 [39:32]	RX_ADDR_P1 [31:24]	RX_ADDR_P1 [23:16]	RX_ADDR_P1 [15:8]	RX_ADDR_P4 [7:0]
R/W	R/W	R/W	R/W	R/W	R/W
POR	C2	C2	C2	C2	C5

Bit 39~8 **RX_ADDR_P1 [39:8]:** Data pipe1 receive address bit 39~bit 8

Bit 7~0 **RX_ADDR_P4 [7:0]:** Data pipe 4 receive address bit 7~bit 0

This field is used to define the data pipe 4 receive address. Note that only bit 7~bit 0 of the address field can be configured and other 32MSBs from bit 39 to bit 8 must be the same as the data pipe 1 address.

- **Address 0FH – RX_ADDR_P5 Register**

This register is used to specify the data pipe 5 RX address.

Byte	4	3	2	1	0
Name	RX_ADDR_P1 [39:32]	RX_ADDR_P1 [31:24]	RX_ADDR_P1 [23:16]	RX_ADDR_P1 [15:8]	RX_ADDR_P5 [7:0]
R/W	R/W	R/W	R/W	R/W	R/W
POR	C2	C2	C2	C2	C6

Bit 39~8 **RX_ADDR_P1 [39:8]:** Data pipe1 receive address bit 39~bit 8

Bit 7~0 **RX_ADDR_P5 [7:0]:** Data pipe 5 receive address bit 7~bit 0

This field is used to define the data pipe 5 receive address. Note that only bit 7~bit 0 of the address field can be configured and other 32MSBs from bit 39 to bit 8 must be the same as the data pipe 1 address.

- **Address 10H – TX_ADDR Register**

This register is used to specify the TX address.

Byte	4	3	2	1	0
Name	TX_ADDR [39:32]	TX_ADDR [31:24]	TX_ADDR [23:16]	TX_ADDR [15:8]	TX_ADDR [7:0]
R/W	R/W	R/W	R/W	R/W	R/W
POR	E7	E7	E7	E7	E7

Bit 39~0 **TX_ADDR:** TX Transmit address

This field is only used for the PTX device to define the TX transmit address. The address field can be up to 5 bytes which is specified in the SETUP_AW register. This address field configuration is carried out in a specific way from the least significant byte to the most significant byte. It is recommended to specify the same address for both RX_ADDR_P0 and TX_ADDR address field to properly handle the automatic acknowledgement function.

- **Address 11H – RX_PW_P0 Register**

This register is used to specify the data pipe 0 RX payload byte number.

Bit	7	6	5	4	3	2	1	0
Name	—	—	RX_PW_ P05	RX_PW_ P04	RX_PW_ P03	RX_PW_ P02	RX_PW_ P01	RX_PW_ P00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 Reserved bits, only data “00” allowed.

Bit 5~0 **RX_PW_P0 [5:0]:** Data Pipe 0 RX payload byte number
0: Not used
1: 1 byte
2: 2 bytes
:
32: 32 bytes
Others: Can not be used

- **Address 12H – RX_PW_P1 Register**

This register is used to specify the data pipe 1 RX payload byte number.

Bit	7	6	5	4	3	2	1	0
Name	—	—	RX_PW_ P15	RX_PW_ P14	RX_PW_ P13	RX_PW_ P12	RX_PW_ P11	RX_PW_ P10
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 Reserved bits, only data “00” allowed.

Bit 5~0 **RX_PW_P1 [5:0]:** Data Pipe 1 RX payload byte number
0: Not used
1: 1 byte
2: 2 bytes
:
32: 32 bytes
Others: Can not be used

- **Address 13H – RX_PW_P2 Register**

This register is used to specify the data pipe 2 RX payload byte number.

Bit	7	6	5	4	3	2	1	0
Name	—	—	RX_PW_ P25	RX_PW_ P24	RX_PW_ P23	RX_PW_ P22	RX_PW_ P21	RX_PW_ P20
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 Reserved bits, only data “00” allowed.

Bit 5~0 **RX_PW_P2 [5:0]:** Data Pipe 2 RX payload byte number
0: Not used
1: 1 byte
2: 2 bytes
:
32: 32 bytes
Others: Can not be used

- **Address 14H – RX_PW_P3 Register**

This register is used to specify the data pipe 3 RX payload byte number.

Bit	7	6	5	4	3	2	1	0
Name	—	—	RX_PW_P35	RX_PW_P34	RX_PW_P33	RX_PW_P32	RX_PW_P31	RX_PW_P30
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 Reserved bits, only data “00” allowed.

Bit 5~0 **RX_PW_P3 [5:0]:** Data Pipe 3 RX payload byte number
0: Not used
1: 1 byte
2: 2 bytes
:
32: 32 bytes
Others: Can not be used.

- **Address 15H – RX_PW_P4 Register**

This register is used to specify the data pipe 4 RX payload byte number.

Bit	7	6	5	4	3	2	1	0
Name	—	—	RX_PW_P45	RX_PW_P44	RX_PW_P43	RX_PW_P42	RX_PW_P41	RX_PW_P40
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 Reserved bits, only data “00” allowed.

Bit 5~0 **RX_PW_P4 [5**:0]: Data Pipe 4 RX payload byte number
0: Not used
1: 1 byte
2: 2 bytes
:
32: 32 bytes
Others: Can not be used.

- **Address 16H – RX_PW_P5 Register**

This register is used to specify the data pipe 5 RX payload byte number.

Bit	7	6	5	4	3	2	1	0
Name	—	—	RX_PW_P55	RX_PW_P54	RX_PW_P53	RX_PW_P52	RX_PW_P51	RX_PW_P50
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 Reserved bits, only data “00” allowed.

Bit 5~0 **RX_PW_P5 [5:0]:** Data Pipe 5 RX payload byte number
0: Not used
1: 1 byte
2: 2 bytes
:
32: 32 bytes
Others: Can not be used.

- **Address 17H – FIFO_STATUS Register**

This register is used to indicate the FIFO status during the data transfer.

Bit	7	6	5	4	3	2	1	0
Name	—	TX_REUSE	TX_FULL	TX_EMPTY	—	—	RX_FULL	RX_EMPTY
R/W	R/W	R	R	R	R/W	R/W	R	R
POR	0	0	0	0	0	0	0	1

Bit 7 Reserved bits, only data “0” allowed.

Bit 6 **TX_REUSE:** Last Transmitted Data Packet Reuse status
0: Last transmitted data packet is not reused
1: Last transmitted data packet is reused

This bit is used to indicate whether the last TX data packet is repeatedly transmitted or not. If the CE line is kept in a high state, the last data packet will be repeatedly retransmitted. This bit is set to 1 by the SPI command, REUSE_TX_PL, and cleared to 0 by the SPI command, W_TX_PAYLOAD or FLUSH_TX.

Bit 5 **TX_FULL:** TX FIFO Full Flag
0: TX FIFO is not full
1: TX FIFO is full

This bit is used to indicate whether the TX FIFO is full or not.

Bit 4 **TX_EMPTY:** TX FIFO Empty Flag
0: TX FIFO is not empty
1: TX FIFO is empty

This bit is used to indicate whether the TX FIFO is empty or not.

Bit 3~2 Reserved bits, only data “00” allowed.

Bit 1 **RX_FULL:** RX FIFO Full Flag
0: RX FIFO is not full
1: RX FIFO is full

This bit is used to indicate whether the RX FIFO is full or not.

Bit 0 **RX_EMPTY:** RX FIFO Empty Flag
0: RX FIFO is not empty
1: RX FIFO is empty

This bit is used to indicate whether the RX FIFO is empty or not.

- **Address 1CH – DYNPD Register**

This register is used to control the individual data pipe dynamic payload length function.

Bit	7	6	5	4	3	2	1	0
Name	—	—	DPL_P5	DPL_P4	DPL_P3	DPL_P2	DPL_P1	DPL_P0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 Reserved bits, only data “00” allowed.

Bit 5 **DPL_P5:** Data Pipe 5 Dynamic Payload Length function enable control
0: Disable
1: Enable

This bit is used to control the data pipe5 dynamic payload length function. It is only available when the EN_DPL and ENAA_P5 bits are both set to 1.

Bit 4 **DPL_P4:** Data Pipe 4 Dynamic Payload Length function enable control
0: Disable
1: Enable

This bit is used to control the data pipe4 dynamic payload length function. It is only available when the EN_DPL and ENAA_P4 bits are both set to 1.

Bit 3 **DPL_P3:** Data Pipe 3 Dynamic Payload Length function enable control
0: Disable
1: Enable

This bit is used to control the data pipe3 dynamic payload length function. It is only available when the EN_DPL and ENAA_P3 bits are both set to 1.

Bit 2 **DPL_P2:** Data Pipe 2 Dynamic Payload Length function enable control
0: Disable
1: Enable

This bit is used to control the data pipe2 dynamic payload length function. It is only available when the EN_DPL and ENAA_P2 bits are both set to 1.

Bit 1 **DPL_P1:** Data Pipe 1 Dynamic Payload Length function enable control
0: Disable
1: Enable

This bit is used to control the data pipe1 dynamic payload length function. It is only available when the EN_DPL and ENAA_P1 bits are both set to 1.

Bit 0 **DPL_P0:** Data Pipe 0 Dynamic Payload Length function enable control
0: Disable
1: Enable

This bit is used to control the data pipe0 dynamic payload length function. It is only available when the EN_DPL and ENAA_P0 bits are both set to 1.

- **Address 1DH – FEATURE Register**

This register is used to control several main features of the RF transceiver.

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	EN_DPL	EN_ACK_PAY	EN_DNY_ACK
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~3 Reserved bits, only data "00000" allowed.

Bit 2 **EN_DPL:** RF Transceiver Dynamic Payload Length Function control
0: Disable
1: Enable

This bit is used to control the RF Transceiver dynamic payload length function. If the EN_DPL bit is set to 1 and the corresponding ENAA_Pn bit is also set to 1, the dynamic payload length function of the relevant data pipe n will be available.

Bit 1 **EN_ACK_PAY:** RF Transceiver Payload with Acknowledgement control
0: Disable
1: Enable

Bit 0 **EN_DNY_ACK:** RF Transceiver "W_TX_PAYLOAD_NOACK" Command control
0: Disable the command
1: Enable the command

Register Bank 1

It is recommended that no access is executed on reserved or non defined registers. Otherwise, this may result in unpredictable conditions.

- **Address 00H – Optimisation Register 0**

Bit	31	30	…………	1	0
Name	Optimisation_Value_0				
R/W	W				
POR	0x0000_0000				

This register is used to optimise the performance of the RF Transceiver by writing a specific value of 0x858A_C01C into this register.

- **Address 01H – Optimisation Register 1**

Bit	31	30	…………	1	0
Name	Optimisation_Value_1				
R/W	W				
POR	0x0000_0000				

This register is used to optimise the performance of the RF Transceiver by writing a specific value of 0x1103_C960 into this register.

- **Address 02H – Optimisation Register 2**

Bit	31	30	…………	1	0
Name	Optimisation_Value_2				
R/W	W				
POR	0x0000_0000				

This register is used to optimise the performance of the RF Transceiver by writing a specific value of 0x0000_0004 into this register.

- **Address 03H – Optimisation Register 3**

Bit	31	30	…………	1	0
Name	Optimisation_Value_3				
R/W	W				
POR	0x0300_1200				

This register is used to optimise the performance of the RF Transceiver by writing a specific value of 0x0000_0004 into this register.

- **Address 04H – Optimisation Register 4**

Bit	31	30	…………	1	0
Name	Optimisation_Value_4				
R/W	W				
POR	0x0000_0000				

This register is used to optimise the performance of the RF Transceiver by writing a specific value into this register for different operating modes.

- For 1Mbps and 2Mbps: 0x437D_5D5F
- For 250kbps: 0x437D_6D5F

For single carrier mode:

- For 1Mbps and 2Mbps: 0xC37D_5D5D
- For 250kbps: 0xC37D_6D5D

- **Address 05H – Optimisation Register 5**

Bit	31	30	…………	1	0
Name	Optimisation_Value_5				
R/W	W				
POR	0x0000_0000				

This register is used to optimise the performance of the RF Transceiver by writing a specific value into this register for different data rates.

- For 250kbps data rate: 0x7410_6C9F
- For 1Mbps data rate: 0x1412_6C9F
- For 2Mbps data rate: 0x7411_4C9F

- **Address 06H – Optimisation Register 6**

Bit	31	30	………	1	0
Name	Optimisation Value 6				
R/W	W				
POR	0x0000_0000				

This register is used to optimise the performance of the RF Transceiver by writing a specific value into this register.

- For 1Mbps: 0x0007_C022
- For 250kbps and 2Mbps: 0x0007_C002

- **Address 07H – Status Register 1**

Bit	31	30		8	7	6	....	1	0
Name	Reserved bits				RBANK	Reserved bits			
R/W	W				R	W			
POR	0				0	0			

This register is reserved except for bit 7, RBANK. It is recommended the reserved bits are not accessed to prevent unpredictable results.

Bit 7 **RBANK:** Register Bank Status
0: Register bank 0
1: Register bank 1

- **Address 08H – Chip ID Register**

Bit	31	30	....	1	0
Name	ID_Value				
R/W	R				
POR	0x0000_0000				

This register is a read-only register and is used to store the chip ID code.

- **Address 0CH – Initialization Register**

Bit	31	30	....	1	0
Name	Initialisation_Value				
R/W	W				
POR	0x0000_0000				

This register is used to initialise the RF Transceiver PLL Lock time by writing a specific value of 0x0573_1200 into this register.

- **Address 0DH – New_Feature Register**

Bit	31	30	....	1	0
Name	New_Feature_Value				
R/W	W				
POR	0x0000_0000				

This register is used to configure the RF Transceiver features by writing a specific value into this register.

- For 1Mbps and 2Mbps: 0x0080_B434
- For 250kbps: 0x0080_B436

- **Address 0EH – RAMP Register**

Bit	87	86	....	1	0
Name	RAMP_Value				
R/W	W				
POR	0xuu_uuuu_ uuuu_uuuu_ uuuu_uuuu				

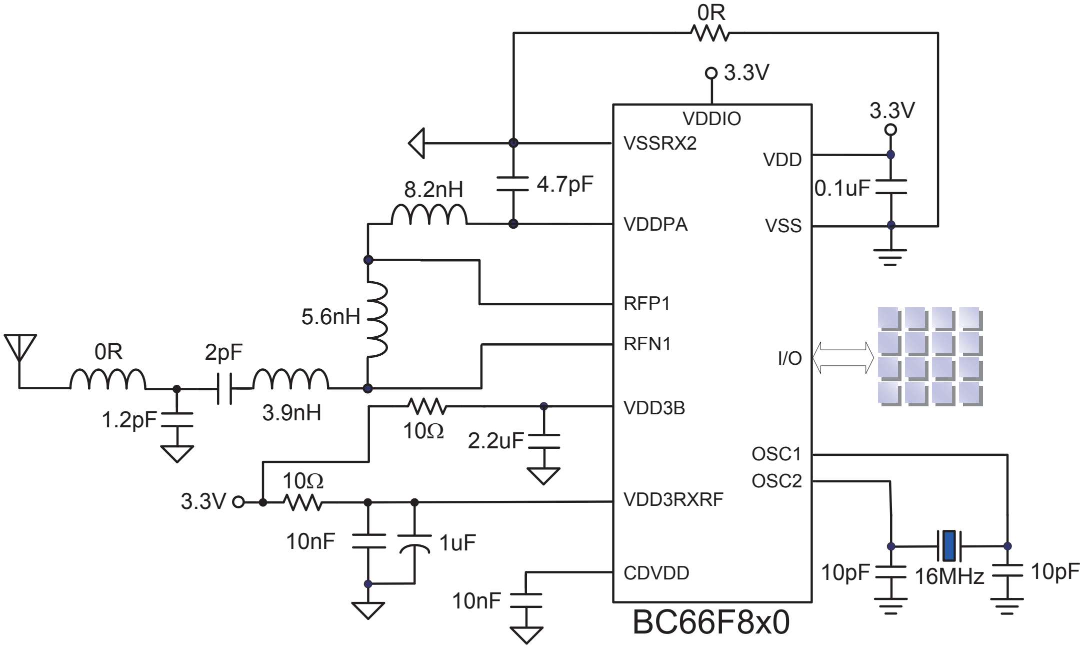
This register is used to optimise the RF Transmitter output spectrum rate curve by writing a specific value of 0xCF_EFBE_F7DF_7CF3_CF20_8104 into this register.

Configuration Options

Configuration options refer to certain options within the MCU that are programmed into the devices during the programming process. During the development process, these options are selected using the HT-IDE software development tools. As these options are programmed into the devices using the hardware programming tools, once they are selected they cannot be changed later using the application program. All options must be defined for proper system function, the details of which are shown in the table.

No.	Options
Oscillator Options	
1	Supplementary Oscillator Selection – f_{SUB}: LXT or LIRC
I/O Pin-Shared Options	
2	Reset pin function: $\overline{RES}$ or I/O
3	VDDIO pin function: VDDIO or I/O
SIM/SPIA Interface Options	
4	SIM Function: Enable or Disable
5	SPIA Function: Enable or Disable
6	SPI/SPIA – CSEN bit: Enable or Disable
7	SPI/SPIA – WCOL bit: Enable or Disable
8	I^2C Debounce Time Selection: No debounce, 2 system clock debounce or 4 system clock debounce

Application Circuits



Instruction Set

Introduction

Central to the successful operation of any microcontroller is its instruction set, which is a set of program instruction codes that directs the microcontroller to perform certain operations. In the case of Holtek microcontroller, a comprehensive and flexible set of over 60 instructions is provided to enable programmers to implement their application with the minimum of programming overheads.

For easier understanding of the various instruction codes, they have been subdivided into several functional groupings.

Instruction Timing

Most instructions are implemented within one instruction cycle. The exceptions to this are branch, call, or table read instructions where two instruction cycles are required. One instruction cycle is equal to 4 system clock cycles, therefore in the case of an 8MHz system oscillator, most instructions would be implemented within 0.5μs and branch or call instructions would be implemented within 1μs. Although instructions which require one more cycle to implement are generally limited to the JMP, CALL, RET, RETI and table read instructions, it is important to realize that any other instructions which involve manipulation of the Program Counter Low register or PCL will also take one more cycle to implement. As instructions which change the contents of the PCL will imply a direct jump to that new address, one more cycle will be required. Examples of such instructions would be “CLR PCL” or “MOV PCL, A”. For the case of skip instructions, it must be noted that if the result of the comparison involves a skip operation then this will also take one more cycle, if no skip is involved then only one cycle is required.

Moving and Transferring Data

The transfer of data within the microcontroller program is one of the most frequently used operations. Making use of three kinds of MOV instructions, data can be transferred from registers to the Accumulator and vice-versa as well as being able to move specific immediate data directly into the Accumulator. One of the most important data transfer applications is to receive data from the input ports and transfer data to the output ports.

Arithmetic Operations

The ability to perform certain arithmetic operations and data manipulation is a necessary feature of most microcontroller applications. Within the Holtek microcontroller instruction set are a range of add and subtract instruction mnemonics to enable the necessary arithmetic to be carried out. Care must be taken to ensure correct handling of carry and borrow data when results exceed 255 for addition and less than 0 for subtraction. The increment and decrement instructions INC, INCA, DEC and DECA provide a simple means of increasing or decreasing by a value of one of the values in the destination specified.

Logical and Rotate Operation

The standard logical operations such as AND, OR, XOR and CPL all have their own instruction within the Holtek microcontroller instruction set. As with the case of most instructions involving data manipulation, data must pass through the Accumulator which may involve additional programming steps. In all logical data operations, the zero flag may be set if the result of the operation is zero. Another form of logical data manipulation comes from the rotate instructions such as RR, RL, RRC and RLC which provide a simple means of rotating one bit right or left. Different rotate instructions exist depending on program requirements. Rotate instructions are useful for serial port programming applications where data can be rotated from an internal register into the Carry bit from where it can be examined and the necessary serial bit set high or low. Another application which rotate data operations are used is to implement multiplication and division calculations.

Branches and Control Transfer

Program branching takes the form of either jumps to specified locations using the JMP instruction or to a subroutine using the CALL instruction. They differ in the sense that in the case of a subroutine call, the program must return to the instruction immediately when the subroutine has been carried out. This is done by placing a return instruction “RET” in the subroutine which will cause the program to jump back to the address right after the CALL instruction. In the case of a JMP instruction, the program simply jumps to the desired location. There is no requirement to jump back to the original jumping off point as in the case of the CALL instruction. One special and extremely useful set of branch instructions are the conditional branches. Here a decision is first made regarding the condition of a certain data memory or individual bits. Depending upon the conditions, the program will continue with the next instruction or skip over it and jump to the following instruction. These instructions are the key to decision making and branching within the program perhaps determined by the condition of certain input switches or by the condition of internal data bits.

Bit Operations

The ability to provide single bit operations on Data Memory is an extremely flexible feature of all Holtek microcontrollers. This feature is especially useful for output port bit programming where individual bits or port pins can be directly set high or low using either the “SET [m].i” or “CLR [m].i” instructions respectively. The feature removes the need for programmers to first read the 8-bit output port, manipulate the input data to ensure that other bits are not changed and then output the port with the correct new data. This read-modify-write process is taken care of automatically when these bit operation instructions are used.

Table Read Operations

Data storage is normally implemented by using registers. However, when working with large amounts of fixed data, the volume involved often makes it inconvenient to store the fixed data in the Data Memory. To overcome this problem, Holtek microcontrollers allow an area of Program Memory to be setup as a table where data can be directly stored. A set of easy to use instructions provides the means by which this fixed data can be referenced and retrieved from the Program Memory.

Other Operations

In addition to the above functional instructions, a range of other instructions also exist such as the “HALT” instruction for Power-down operations and instructions to control the operation of the Watchdog Timer for reliable program operations under extreme electric or electromagnetic environments. For their relevant operations, refer to the functional related sections.

Instruction Set Summary

The following table depicts a summary of the instruction set categorised according to function and can be consulted as a basic instruction reference using the following listed conventions.

Table Conventions

x: Bits immediate data
m: Data Memory address
A: Accumulator
i: 0~7 number of bits
addr: Program memory address

Mnemonic	Description	Cycles	Flag Affected
Arithmetic			
ADD A,[m]	Add Data Memory to ACC	1	Z, C, AC, OV
ADDM A,[m]	Add ACC to Data Memory	1^{Note}	Z, C, AC, OV
ADD A,x	Add immediate data to ACC	1	Z, C, AC, OV
ADC A,[m]	Add Data Memory to ACC with Carry	1	Z, C, AC, OV
ADCM A,[m]	Add ACC to Data memory with Carry	1^{Note}	Z, C, AC, OV
SUB A,x	Subtract immediate data from the ACC	1	Z, C, AC, OV
SUB A,[m]	Subtract Data Memory from ACC	1	Z, C, AC, OV
SUBM A,[m]	Subtract Data Memory from ACC with result in Data Memory	1^{Note}	Z, C, AC, OV
SBC A,[m]	Subtract Data Memory from ACC with Carry	1	Z, C, AC, OV
SBCM A,[m]	Subtract Data Memory from ACC with Carry, result in Data Memory	1^{Note}	Z, C, AC, OV
DAA [m]	Decimal adjust ACC for Addition with result in Data Memory	1^{Note}	C
Logic Operation			
AND A,[m]	Logical AND Data Memory to ACC	1	Z
OR A,[m]	Logical OR Data Memory to ACC	1	Z
XOR A,[m]	Logical XOR Data Memory to ACC	1	Z
ANDM A,[m]	Logical AND ACC to Data Memory	1^{Note}	Z
ORM A,[m]	Logical OR ACC to Data Memory	1^{Note}	Z
XORM A,[m]	Logical XOR ACC to Data Memory	1^{Note}	Z
AND A,x	Logical AND immediate Data to ACC	1	Z
OR A,x	Logical OR immediate Data to ACC	1	Z
XOR A,x	Logical XOR immediate Data to ACC	1	Z
CPL [m]	Complement Data Memory	1^{Note}	Z
CPLA [m]	Complement Data Memory with result in ACC	1	Z
Increment & Decrement			
INCA [m]	Increment Data Memory with result in ACC	1	Z
INC [m]	Increment Data Memory	1^{Note}	Z
DECA [m]	Decrement Data Memory with result in ACC	1	Z
DEC [m]	Decrement Data Memory	1^{Note}	Z
Rotate			
RRA [m]	Rotate Data Memory right with result in ACC	1	None
RR [m]	Rotate Data Memory right	1^{Note}	None
RRCA [m]	Rotate Data Memory right through Carry with result in ACC	1	C
RRC [m]	Rotate Data Memory right through Carry	1^{Note}	C
RLA [m]	Rotate Data Memory left with result in ACC	1	None
RL [m]	Rotate Data Memory left	1^{Note}	None
RLCA [m]	Rotate Data Memory left through Carry with result in ACC	1	C
RLC [m]	Rotate Data Memory left through Carry	1^{Note}	C

Mnemonic	Description	Cycles	Flag Affected
Data Move			
MOV A,[m]	Move Data Memory to ACC	1	None
MOV [m],A	Move ACC to Data Memory	1^{Note}	None
MOV A,x	Move immediate data to ACC	1	None
Bit Operation			
CLR [m].i	Clear bit of Data Memory	1^{Note}	None
SET [m].i	Set bit of Data Memory	1^{Note}	None
Branch			
JMP addr	Jump unconditionally	2	None
SZ [m]	Skip if Data Memory is zero	1^{Note}	None
SZA [m]	Skip if Data Memory is zero with data movement to ACC	1^{Note}	None
SZ [m].i	Skip if bit i of Data Memory is zero	1^{Note}	None
SNZ [m].i	Skip if bit i of Data Memory is not zero	1^{Note}	None
SIZ [m]	Skip if increment Data Memory is zero	1^{Note}	None
SDZ [m]	Skip if decrement Data Memory is zero	1^{Note}	None
SIZA [m]	Skip if increment Data Memory is zero with result in ACC	1^{Note}	None
SDZA [m]	Skip if decrement Data Memory is zero with result in ACC	1^{Note}	None
CALL addr	Subroutine call	2	None
RET	Return from subroutine	2	None
RET A,x	Return from subroutine and load immediate data to ACC	2	None
RETI	Return from interrupt	2	None
Table Read			
TABRD [m]	Read table to TBLH and Data Memory	2^{Note}	None
TABRDL [m]	Read table (last page) to TBLH and Data Memory	2^{Note}	None
Miscellaneous			
NOP	No operation	1	None
CLR [m]	Clear Data Memory	1^{Note}	None
SET [m]	Set Data Memory	1^{Note}	None
CLR WDT	Clear Watchdog Timer	1	TO, PDF
CLR WDT1	Pre-clear Watchdog Timer	1	TO, PDF
CLR WDT2	Pre-clear Watchdog Timer	1	TO, PDF
SWAP [m]	Swap nibbles of Data Memory	1^{Note}	None
SWAPA [m]	Swap nibbles of Data Memory with result in ACC	1	None
HALT	Enter power down mode	1	TO, PDF

Note: 1. For skip instructions, if the result of the comparison involves a skip then two cycles are required, if no skip takes place only one cycle is required.

2. Any instruction which changes the contents of the PCL will also require 2 cycles for execution.

3. For the “CLR WDT1” and “CLR WDT2” instructions the TO and PDF flags may be affected by the execution status. The TO and PDF flags are cleared after both “CLR WDT1” and “CLR WDT2” instructions are consecutively executed. Otherwise the TO and PDF flags remain unchanged.

Instruction Definition

ADC A,[m] Add Data Memory to ACC with Carry

Description The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the Accumulator.

Operation ACC ← ACC + [m] + C

Affected flag(s) OV, Z, AC, C

ADCM A,[m] Add ACC to Data Memory with Carry

Description The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the specified Data Memory.

Operation [m] ← ACC + [m] + C

Affected flag(s) OV, Z, AC, C

ADD A,[m] Add Data Memory to ACC

Description The contents of the specified Data Memory and the Accumulator are added. The result is stored in the Accumulator.

Operation ACC ← ACC + [m]

Affected flag(s) OV, Z, AC, C

ADD A,x Add immediate data to ACC

Description The contents of the Accumulator and the specified immediate data are added. The result is stored in the Accumulator.

Operation ACC ← ACC + x

Affected flag(s) OV, Z, AC, C

ADDM A,[m] Add ACC to Data Memory

Description The contents of the specified Data Memory and the Accumulator are added. The result is stored in the specified Data Memory.

Operation [m] ← ACC + [m]

Affected flag(s) OV, Z, AC, C

AND A,[m] Logical AND Data Memory to ACC

Description Data in the Accumulator and the specified Data Memory perform a bitwise logical AND operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″AND″ [m]

Affected flag(s) Z

AND A,x Logical AND immediate data to ACC

Description Data in the Accumulator and the specified immediate data perform a bit wise logical AND operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″AND″ x

Affected flag(s) Z

ANDM A,[m] Logical AND ACC to Data Memory

Description Data in the specified Data Memory and the Accumulator perform a bitwise logical AND operation. The result is stored in the Data Memory.

Operation [m] ← ACC ″AND″ [m]

Affected flag(s) Z

CALL addr Subroutine call

Description Unconditionally calls a subroutine at the specified address. The Program Counter then increments by 1 to obtain the address of the next instruction which is then pushed onto the stack. The specified address is then loaded and the program continues execution from this new address. As this instruction requires an additional operation, it is a two cycle instruction.

Operation Stack ← Program Counter + 1
Program Counter ← addr

Affected flag(s) None

CLR [m] Clear Data Memory

Description Each bit of the specified Data Memory is cleared to 0.

Operation [m] ← 00H

Affected flag(s) None

CLR [m].i Clear bit of Data Memory

Description Bit i of the specified Data Memory is cleared to 0.

Operation [m].i ← 0

Affected flag(s) None

CLR WDT Clear Watchdog Timer

Description The TO, PDF flags and the WDT are all cleared.

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CLR WDT1 Pre-clear Watchdog Timer

Description The TO, PDF flags and the WDT are all cleared. Note that this instruction works in conjunction with CLR WDT2 and must be executed alternately with CLR WDT2 to have effect. Repetitively executing this instruction without alternately executing CLR WDT2 will have no effect.

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CLR WDT2 Pre-clear Watchdog Timer

Description The TO, PDF flags and the WDT are all cleared. Note that this instruction works in conjunction with CLR WDT1 and must be executed alternately with CLR WDT1 to have effect. Repetitively executing this instruction without alternately executing CLR WDT1 will have no effect.

Operation WDT cleared
TO ← 0
PDF ← 0

Affected flag(s) TO, PDF

CPL [m] Complement Data Memory

Description Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa.

Operation [m] ← $\overline{[m]}$

Affected flag(s) Z

CPLA [m] Complement Data Memory with result in ACC

Description Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa. The complemented result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC ← $\overline{[m]}$

Affected flag(s) Z

DAA [m] Decimal-Adjust ACC for addition with result in Data Memory

Description Convert the contents of the Accumulator value to a BCD (Binary Coded Decimal) value resulting from the previous addition of two BCD variables. If the low nibble is greater than 9 or if AC flag is set, then a value of 6 will be added to the low nibble. Otherwise the low nibble remains unchanged. If the high nibble is greater than 9 or if the C flag is set, then a value of 6 will be added to the high nibble. Essentially, the decimal conversion is performed by adding 00H, 06H, 60H or 66H depending on the Accumulator and flag conditions. Only the C flag may be affected by this instruction which indicates that if the original BCD sum is greater than 100, it allows multiple precision decimal addition.

Operation [m] ← ACC + 00H or
[m] ← ACC + 06H or
[m] ← ACC + 60H or
[m] ← ACC + 66H

Affected flag(s) C

DEC [m] Decrement Data Memory

Description Data in the specified Data Memory is decremented by 1.

Operation [m] ← [m] − 1

Affected flag(s) Z

DECA [m] Decrement Data Memory with result in ACC

Description Data in the specified Data Memory is decremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC ← [m] − 1

Affected flag(s) Z

HALT Enter power down mode

Description This instruction stops the program execution and turns off the system clock. The contents of the Data Memory and registers are retained. The WDT and prescaler are cleared. The power down flag PDF is set and the WDT time-out flag TO is cleared.

Operation TO ← 0
PDF ← 1

Affected flag(s) TO, PDF

INC [m] Increment Data Memory

Description Data in the specified Data Memory is incremented by 1.

Operation [m] ← [m] + 1

Affected flag(s) Z

INCA [m] Increment Data Memory with result in ACC

Description Data in the specified Data Memory is incremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC ← [m] + 1

Affected flag(s) Z

JMP addr Jump unconditionally

Description The contents of the Program Counter are replaced with the specified address. Program execution then continues from this new address. As this requires the insertion of a dummy instruction while the new address is loaded, it is a two cycle instruction.

Operation Program Counter ← addr

Affected flag(s) None

MOV A,[m] Move Data Memory to ACC

Description The contents of the specified Data Memory are copied to the Accumulator.

Operation ACC ← [m]

Affected flag(s) None

MOV A,x Move immediate data to ACC

Description The immediate data specified is loaded into the Accumulator.

Operation ACC ← x

Affected flag(s) None

MOV [m],A Move ACC to Data Memory

Description The contents of the Accumulator are copied to the specified Data Memory.

Operation [m] ← ACC

Affected flag(s) None

NOP No operation

Description No operation is performed. Execution continues with the next instruction.

Operation No operation

Affected flag(s) None

OR A,[m] Logical OR Data Memory to ACC

Description Data in the Accumulator and the specified Data Memory perform a bitwise logical OR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″OR″ [m]

Affected flag(s) Z

OR A,x Logical OR immediate data to ACC

Description Data in the Accumulator and the specified immediate data perform a bitwise logical OR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″OR″ x

Affected flag(s) Z

ORM A,[m] Logical OR ACC to Data Memory

Description Data in the specified Data Memory and the Accumulator perform a bitwise logical OR operation. The result is stored in the Data Memory.

Operation [m] ← ACC ″OR″ [m]

Affected flag(s) Z

RET Return from subroutine

Description The Program Counter is restored from the stack. Program execution continues at the restored address.

Operation Program Counter ← Stack

Affected flag(s) None

RET A,x Return from subroutine and load immediate data to ACC

Description The Program Counter is restored from the stack and the Accumulator loaded with the specified immediate data. Program execution continues at the restored address.

Operation Program Counter ← Stack
ACC ← x

Affected flag(s) None

RETI Return from interrupt

Description The Program Counter is restored from the stack and the interrupts are re-enabled by setting the EMI bit. EMI is the master interrupt global enable bit. If an interrupt was pending when the RETI instruction is executed, the pending Interrupt routine will be processed before returning to the main program.

Operation Program Counter ← Stack
EMI ← 1

Affected flag(s) None

RL [m] Rotate Data Memory left

Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0.

Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← [m].7

Affected flag(s) None

RLA [m] Rotate Data Memory left with result in ACC

Description The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← [m].7

Affected flag(s) None

RLC [m] Rotate Data Memory left through Carry

Description The contents of the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into bit 0.

Operation [m].(i+1) ← [m].i; (i=0~6)
[m].0 ← C
C ← [m].7

Affected flag(s) C

RLCA [m] Rotate Data Memory left through Carry with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into the bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.(i+1) ← [m].i; (i=0~6)
ACC.0 ← C
C ← [m].7

Affected flag(s) C

RR [m] Rotate Data Memory right

Description The contents of the specified Data Memory are rotated right by 1 bit with bit 0 rotated into bit 7.

Operation [m].i ← [m].(i+1); (i=0~6)
[m].7 ← [m].0

Affected flag(s) None

RRA [m] Rotate Data Memory right with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated right by 1 bit with bit 0 rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.i ← [m].(i+1); (i=0~6)
ACC.7 ← [m].0

Affected flag(s) None

RRC [m] Rotate Data Memory right through Carry

Description The contents of the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7.

Operation [m].i ← [m].(i+1); (i=0~6)
[m].7 ← C
C ← [m].0

Affected flag(s) C

RRCA [m] Rotate Data Memory right through Carry with result in ACC

Description Data in the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation ACC.i ← [m].(i+1); (i=0~6)
ACC.7 ← C
C ← [m].0

Affected flag(s) C

SBC A,[m] Subtract Data Memory from ACC with Carry

Description The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation ACC ← ACC − [m] − C

Affected flag(s) OV, Z, AC, C

SBCM A,[m] Subtract Data Memory from ACC with Carry and result in Data Memory

Description The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation [m] ← ACC − [m] − C

Affected flag(s) OV, Z, AC, C

SDZ [m] Skip if decrement Data Memory is 0

Description The contents of the specified Data Memory are first decremented by 1. If the result is 0 the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation [m] ← [m] − 1
Skip if [m]=0

Affected flag(s) None

SDZA [m] Skip if decrement Data Memory is zero with result in ACC

Description The contents of the specified Data Memory are first decremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0, the program proceeds with the following instruction.

Operation ACC ← [m] − 1
Skip if ACC=0

Affected flag(s) None

SET [m] Set Data Memory

Description Each bit of the specified Data Memory is set to 1.

Operation [m] ← FFH

Affected flag(s) None

SET [m].i Set bit of Data Memory

Description Bit i of the specified Data Memory is set to 1.

Operation [m].i ← 1

Affected flag(s) None

SIZ [m] Skip if increment Data Memory is 0

Description The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation [m] ← [m] + 1
Skip if [m]=0

Affected flag(s) None

SIZA [m] Skip if increment Data Memory is zero with result in ACC

Description The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation ACC ← [m] + 1
Skip if ACC=0

Affected flag(s) None

SNZ [m].i Skip if bit i of Data Memory is not 0

Description If bit i of the specified Data Memory is not 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is 0 the program proceeds with the following instruction.

Operation Skip if [m].i ≠ 0

Affected flag(s) None

SUB A,[m] Subtract Data Memory from ACC

Description The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation ACC ← ACC − [m]

Affected flag(s) OV, Z, AC, C

SUBM A,[m] Subtract Data Memory from ACC with result in Data Memory

Description The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation [m] ← ACC − [m]

Affected flag(s) OV, Z, AC, C

SUB A,x Subtract immediate data from ACC

Description The immediate data specified by the code is subtracted from the contents of the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.

Operation ACC ← ACC − x

Affected flag(s) OV, Z, AC, C

SWAP [m] Swap nibbles of Data Memory

Description The low-order and high-order nibbles of the specified Data Memory are interchanged.

Operation [m].3~[m].0 ↔ [m].7~[m].4

Affected flag(s) None

SWAPA [m] Swap nibbles of Data Memory with result in ACC

Description The low-order and high-order nibbles of the specified Data Memory are interchanged. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.

Operation ACC.3~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0

Affected flag(s) None

SZ [m] Skip if Data Memory is 0

Description If the contents of the specified Data Memory is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation Skip if [m]=0

Affected flag(s) None

SZA [m] Skip if Data Memory is 0 with data movement to ACC

Description The contents of the specified Data Memory are copied to the Accumulator. If the value is zero, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.

Operation ACC ← [m]
Skip if [m]=0

Affected flag(s) None

SZ [m].i Skip if bit i of Data Memory is 0

Description If bit i of the specified Data Memory is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0, the program proceeds with the following instruction.

Operation Skip if [m].i=0

Affected flag(s) None

TABRD [m] Read table (current page) to TBLH and Data Memory

Description The low byte of the program code (current page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

TABRDL [m] Read table (last page) to TBLH and Data Memory

Description The low byte of the program code (last page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.

Operation [m] ← program code (low byte)
TBLH ← program code (high byte)

Affected flag(s) None

XOR A,[m] Logical XOR Data Memory to ACC

Description Data in the Accumulator and the specified Data Memory perform a bitwise logical XOR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″XOR″ [m]

Affected flag(s) Z

XORM A,[m] Logical XOR ACC to Data Memory

Description Data in the specified Data Memory and the Accumulator perform a bitwise logical XOR operation. The result is stored in the Data Memory.

Operation [m] ← ACC ″XOR″ [m]

Affected flag(s) Z

XOR A,x Logical XOR immediate data to ACC

Description Data in the Accumulator and the specified immediate data perform a bitwise logical XOR operation. The result is stored in the Accumulator.

Operation ACC ← ACC ″XOR″ x

Affected flag(s) Z

Package Information

Note that the package information provided here is for consultation purposes only. As this information may be updated at regular intervals users are reminded to consult the Holtek website for the latest version of the Package/Carton Information.

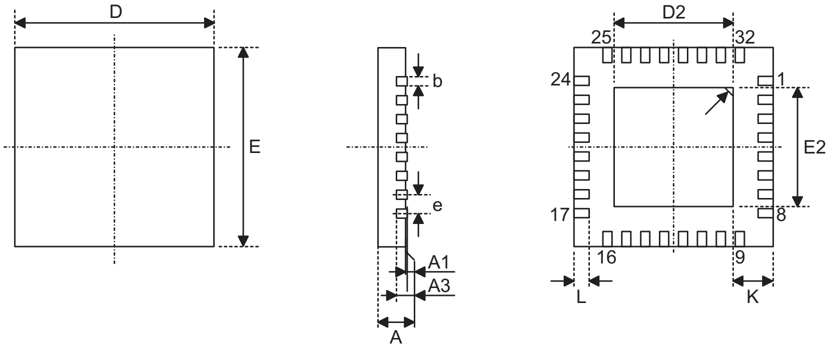
Additional supplementary information with regard to packaging is listed below. Click on the relevant section to be transferred to the relevant website page.

- Further Package Information (include Outline Dimensions, Product Tape and Reel Specifications)

- Packing Meterials Information

- Carton information

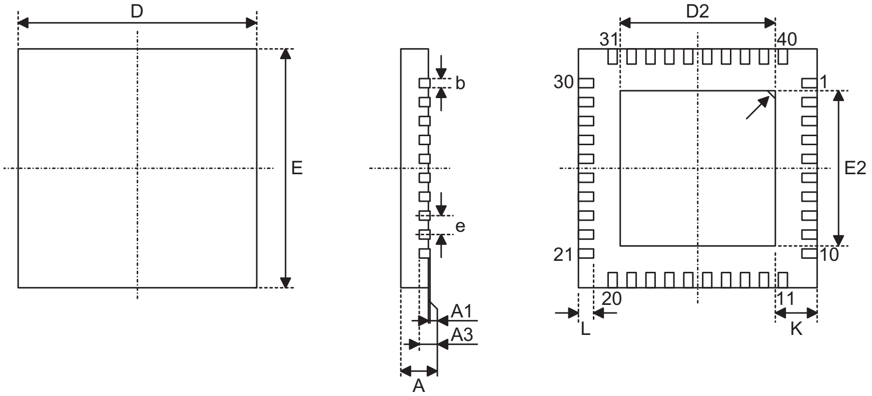
SAW Type 32-pin (5mm×5mm×0.75mm) QFN Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	0.028	0.030	0.031
A1	0.000	0.001	0.002
A3	—	0.008 BSC	—
b	0.007	0.010	0.012
D	0.193	0.197	0.201
E	0.193	0.197	0.201
e	—	0.020 BSC	—
D2	0.122	0.126	0.130
E2	0.122	0.126	0.130
L	0.014	0.016	0.018
K	0.008	—	—

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	0.700	0.750	0.800
A1	0.000	0.020	0.050
A3	—	0.203 BSC	—
b	0.180	0.250	0.300
D	4.900	5.000	5.100
E	4.900	5.000	5.100
e	—	0.50 BSC	—
D2	3.10	3.20	3.30
E2	3.10	3.20	3.30
L	0.35	0.40	0.45
K	0.20	—	—

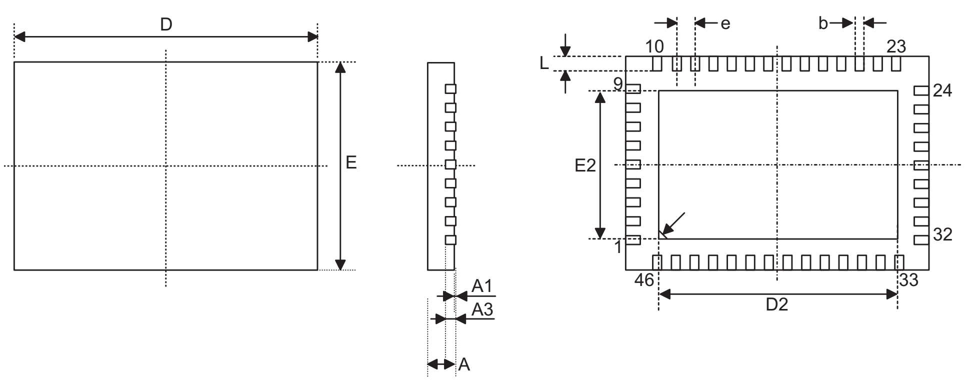
SAW Type 40-pin (6mm×6mm×0.75mm) QFN Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	0.028	0.030	0.031
A1	0.000	0.001	0.002
A3	—	0.008 BSC	—
b	0.007	0.010	0.012
D	0.232	0.236	0.240
E	0.232	0.236	0.240
e	—	0.020 BSC	—
D2	0.173	0.177	0.181
E2	0.173	0.177	0.181
L	0.014	0.016	0.018
K	0.008	—	—

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	0.700	0.750	0.800
A1	0.000	0.020	0.050
A3	—	0.203 BSC	—
b	0.180	0.250	0.300
D	5.900	6.000	6.100
E	5.900	6.000	6.100
e	—	0.50 BSC	—
D2	4.40	4.50	4.60
E2	4.40	4.50	4.60
L	0.35	0.40	0.45
K	0.20	—	—

SAW Type 46-pin (6.5mm×4.5mm×0.85mm) QFN Outline Dimensions



Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	0.031	0.033	0.035
A1	0.000	0.001	0.002
A3	—	0.008 BSC	—
b	0.006	0.008	0.010
D	0.254	0.256	0.258
E	0.175	0.177	0.179
e	—	0.016 BSC	—
D2	0.197	0.201	0.205
E2	0.118	0.122	0.126
L	0.012	0.016	0.020

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	0.800	0.850	0.900
A1	0.000	0.020	0.040
A3	—	0.200 BSC	—
b	0.150	0.200	0.250
D	6.450	6.500	6.550
E	4.450	4.500	4.550
e	—	0.40 BSC	—
D2	5.00	5.10	5.20
E2	3.00	3.10	3.20
L	0.30	0.40	0.50

The information appearing in this Data Sheet is believed to be accurate at the time of publication. However, Holtek assumes no responsibility arising from the use of the specifications described. The applications mentioned herein are used solely for the purpose of illustration and Holtek makes no warranty or representation that such applications will be suitable without further modification, nor recommends the use of its products for application that may present a risk to human life due to malfunction or otherwise. Holtek's products are not authorized for use as critical components in life support devices or systems. Holtek reserves the right to alter its products without prior notification. For the most up-to-date information, please visit our web site at http://www.holtek.com.