十速

TM57FLA80/80A

DATA SHEET

Rev 2.1

tenx reserves the right to change or discontinue the manual and online documentation to this product herein to improve reliability, function or design without further notice. **tenx** does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. **tenx** products are not designed, intended, or authorized for use in life support appliances, devices, or systems. If Buyer purchases or uses tenx products for any such unintended or unauthorized application, Buyer shall indemnify and hold tenx and its officers, employees, subsidiaries, affiliates and distributors harmless against all claims, cost, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use even if such claim alleges that tenx was negligent regarding the design or manufacture of the part.

AMENDMENT HISTORY

Version	Date	Description
V1.3	Dec, 2010	1. Add more description about /Borrow and /Digit Borrow in ALU and Working (W) Register section. 2. Add Internal RC mode description and figure in System Clock Oscillator section. 3. Modify the status affected of the NOP instruction.
V1.4	Apr, 2011	Omit the data in Table 3.8.1 in page 36 and 37.
V1.5	Sept, 2011	1. Revise package information: 44-QFP is modified to 44-LQFP. 2. P62. SLEEP and CLRWDT flag affect 3. P21. Figure 2.4.1 fix 4. P31. Figure 3.6.1 fix, and change PWM0PERIOD to PWM0PRD in text
V1.6	Dec, 2011	Add Ordering Information table in the Packaging Information section.
V1.7	Jan, 2012	1. Add the Electrical Characteristics specs in the Features section. 2. Add description in Reset section. 3. Modify the Low Voltage Detection time data in LVR Circuit Characteristics section.
V1.8	Jul, 2012	Add output current data in Electrical Characteristics section.
V1.9	Aug, 2013	1. Add supported EV board on ICE in Features section. 2. Modify System Block Diagram. 3. Modify Ordering Information. 4. Modify INT0EDGE description.
V2.0	Dec, 2013	1. Modify Package Type 2. Modify QFP-44 Pin Assignment 3. Modify Ordering info 4. Modify Package Dimension
V2.1	Oct, 2014	1. Modify Doc No(TM57FLA80 Modify TM57FLA80&80A) 2. Modify Page Title, Page head/rail 3. Add TM57FLA80A descriptions

CONTENTS

FEATURES

1. **Memory**
   - **176 Bytes on F-plane**
   - **192 Bytes on R-plane**
   - **20 Bytes LCD RAM**
   - **8K x 14 internal flash memory**
2. **Oscillation Sources**
   - **Fast Clock:**
     - Fast XTAL (1~12 MHz) (also denoted by FXT in the text.)
     - FIRC (4 MHz)
     - Fast XRC (also denoted by FXRC in the text.)
   - **Slow Clock:**
     - Slow XTAL (32768 Hz) (also denoted by SXT in the text.)
     - Slow XRC (also denoted by SXRC in the text.)
3. **Instruction Set**
   - 37 instructions
4. **8-level Stack**
5. **Instruction Execution Time**
   - 2 oscillation clocks per instruction except branch
6. **2-Level Low Voltage Reset**
   - 2.1V/2.9V (only support to TM57FLA80)
7. **Operation Voltage: Low Voltage Reset Voltage to 5.5V (for TM57FLA80) , 3.6V (for TM57FLA80A)**
   - fosc = 4 MHz, 2.4V ~ 5.5V
   - fosc = 8 MHz, 2.5V ~ 5.5V
   - fosc = 12 MHz, 3.0V ~ 5.5V
8. **ISP (In-System Programming) uses only 5 wires (VPP, VCC, VSS, PA1, PA0)**
9. **Power Saving Operation Mode**
   - Fast Mode: Slow Clock can be disabled or enabled
   - Slow Mode: Fast Clock stops, CPU is running
   - Idle Mode: Slow Clock is running, CPU stops, LCD can be disabled or enabled, Timer2 is running.
   - Stop Mode: All Clocks stop, Wake-up Timer is disabled or enabled

10. Interrupt

- 9 kinds of interrupt source with individual vector
- 8 External Interrupt pins, 6 pins are falling edge triggered, 2 pins are rising or falling edge triggered.
- Timer0, Timer1, Timer2, Wake-up Timer Interrupt
- PWM interrupt
- UART, SPI interrupt
- INT0, INT1, and INT2 share 1 interrupt vector
- INT3, INT4, INT5, INT6, and INT7 share 1 interrupt vector

11. Automatic Store/Restore W and STATUS when interrupt (register control option)

12. I/O Port

- Maximum 45 programmable I/O pins (48-QFP)
- Pseudo-Open-Drain Output
- Open-Drain Output
- CMOS Push-Pull Output
- Schmitt Trigger Input

13. 3 Independent Timers

- Timer0 is 8-bit with 8-bit prescaler, Counter/Capture/Interrupt function
- Timer1 is 16-bit with Buzzer/Capture/Reload/Interrupt function
- Timer2 is used for LCD clock generation and real time 32768 Hz interrupt

14. 2 Independent 8-bit PWMs

- PWM0 with prescaler/period-adjustment/buffer-reload/rising-falling output
- PWM1 is simple duty controlled PWM

15. Watchdog Timer

- Clocked by on-chip oscillator with 4 adjustable Reset/Interrupt time durations (112 ms / 27.6 ms / 6.68 ms / 3.45 ms)
- Share with Wake-up Timer depends on the System Configuration bit WDTE.

16. 16-channel Touch Key

17. LCD Controller/Driver

- 8 COM X 20 SEG
- 4 COM X 24 SEG
- 3 COM X 24 SEG
- Static
- 1/2, 1/3, 1/4 Bias
- 8 Brightness Levels Selection

18. A/D Converter

- 6 analog input channels
- 12-bit resolutions

19. UART Interface

- 7/8/9 bits mode TX/RX selectable
- Supported Baud-Rate ranges from 1200 bps to 38400 bps with proper selected oscillation frequency and baud rate clock divide.
- Automatic parity generation and detection
- Detects Overrun, Frame Error, and Parity Error

20. SPI Interface

- Master or Slave mode selectable
- Programmable transmit bit rate
- Serial clock phase and polarity options
- nSS (Slave select) output
- MSB-first or LSB-first selectable

21. Operating Temperature Range

- -40℃ to +85℃

22. Package Type

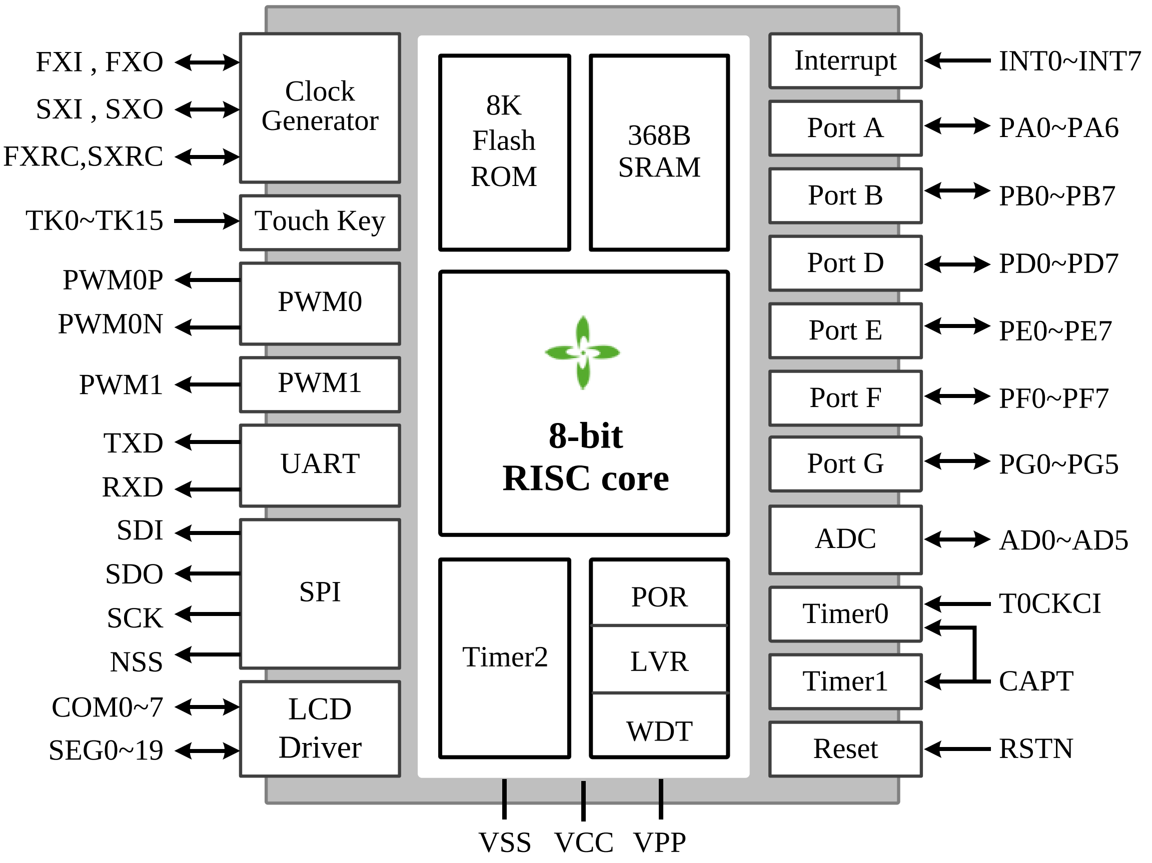
- 48-pin LQFP
- 44-pin QFP
- 32-pin SOP

23. Supported EV board on ICE

EV board: EV2796

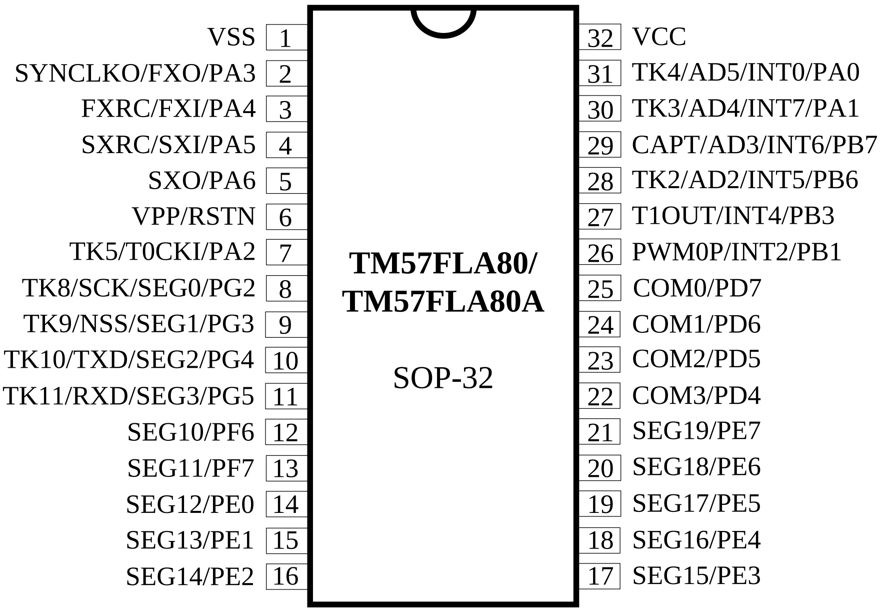
Note that when mention about TM57FLA80, we mean both TM57FLA80 and TM57FLA80A unless specific descriptions.

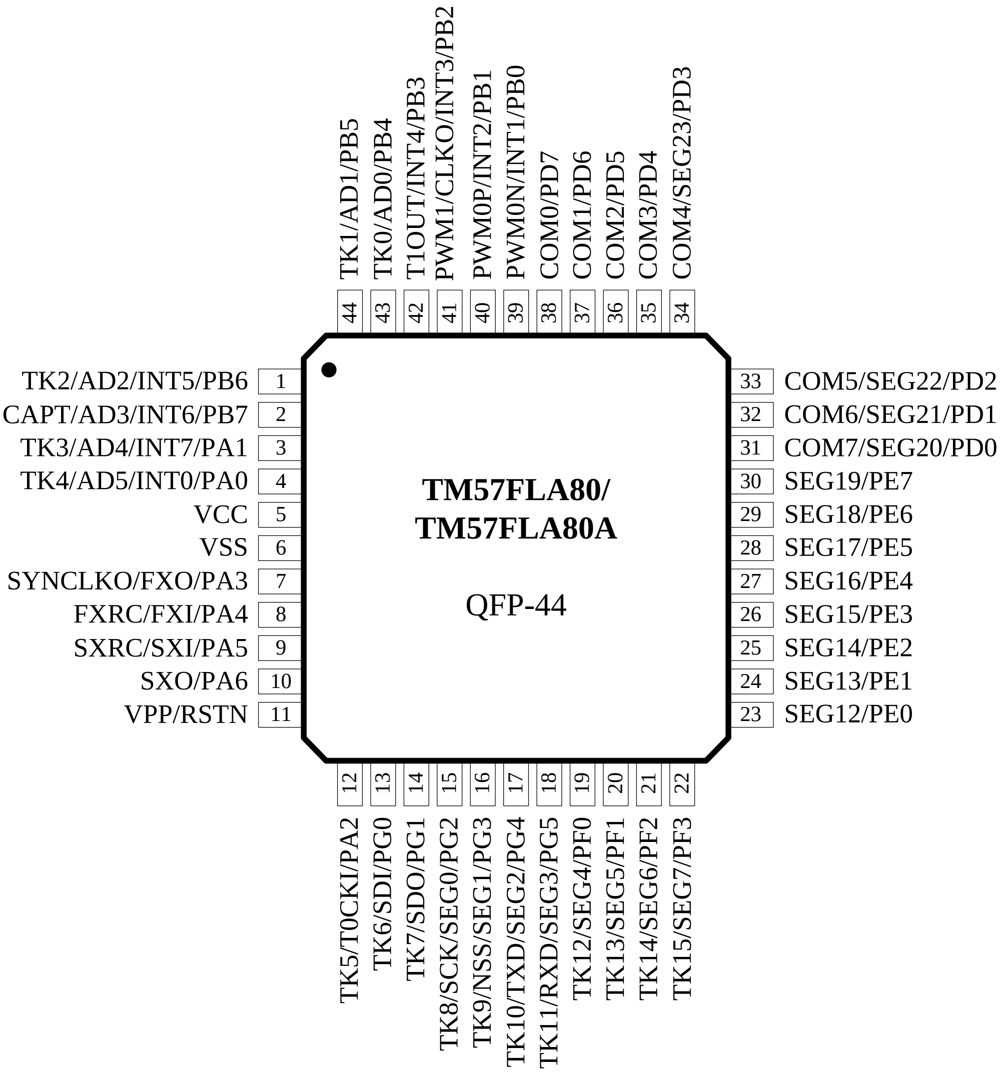
SYSTEM BLOCK DIAGRAM

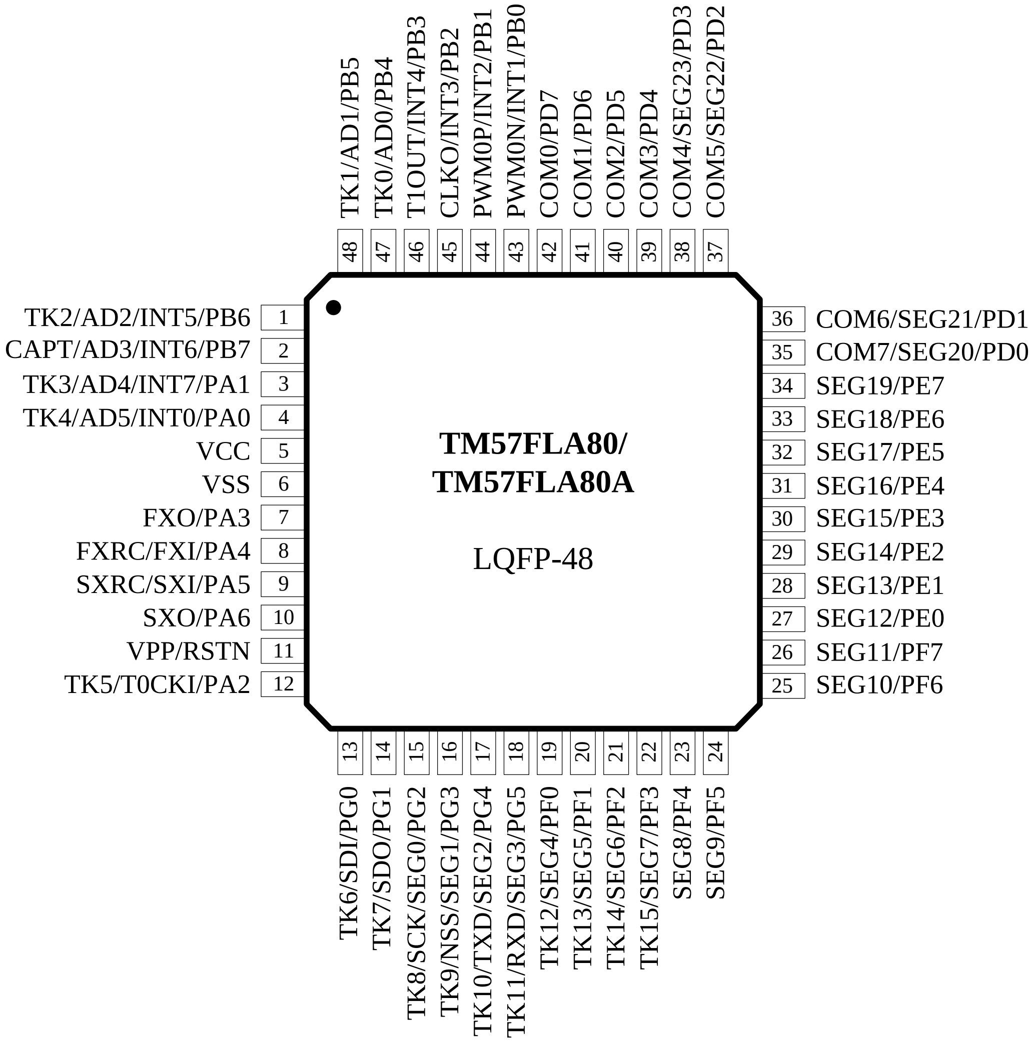


TM57FLA80/80A Block Diagram

PIN ASSIGNMENT DIAGRAM




TM57FLA80/
TM57FLA80A
QFP-44
1 TK2/AD2/INT5/PB6
2 CAPT/AD3/INT6/PB7
3 TK3/AD4/INT7/PA1
4 TK4/AD5/INT0/PA0
5 VCC
6 VSS
7 SYNCLKO/FXO/PA3
8 FXRC/FXI/PA4
9 SXRC/SXI/PA5
10 SXO/PA6
11 VPP/RSTN
12 TK5/T0CKI/PA2
13 TK6/SDI/PG0
14 TK7/SDO/PG1
15 TK8/SCK/SEG0/PG2
16 TK9/NSS/SEG1/PG3
17 TK10/TXD/SEG2/PG4
18 TK11/RXD/SEG3/PG5
19 TK12/SEG4/PF0
20 TK13/SEG5/PF1
21 TK14/SEG6/PF2
22 TK15/SEG7/PF3
23 SEG12/PE0
24 SEG13/PE1
25 SEG14/PE2
26 SEG15/PE3
27 SEG16/PE4
28 SEG17/PE5
29 SEG18/PE6
30 SEG19/PE7
31 COM7/SEG20/PD0
32 COM6/SEG21/PD1
33 COM5/SEG22/PD2
34 COM4/SEG23/PD3
35 COM3/PD4
36 COM2/PD5
37 COM1/PD6
38 COM0/PD7
39 PWM0N/INT1/PB0
40 PWM0P/INT2/PB1
41 PWM1/CLKO/INT3/PB2
42 T1OUT/INT4/PB3
43 TK0/AD0/PB4
44 TK1/AD1/PB5


TM57FLA80/
TM57FLA80A
LQFP-48
1 TK2/AD2/INT5/PB6
2 CAPT/AD3/INT6/PB7
3 TK3/AD4/INT7/PA1
4 TK4/AD5/INT0/PA0
5 VCC
6 VSS
7 FXO/PA3
8 FXRC/FXI/PA4
9 SXRC/SXI/PA5
10 SXO/PA6
11 VPP/RSTN
12 TK5/T0CKI/PA2
13 TK6/SDI/PG0
14 TK7/SDO/PG1
15 TK8/SCK/SEG0/PG2
16 TK9/NSS/SEG1/PG3
17 TK10/TXD/SEG2/PG4
18 TK11/RXD/SEG3/PG5
19 TK12/SEG4/PF0
20 TK13/SEG5/PF1
21 TK14/SEG6/PF2
22 TK15/SEG7/PF3
23 SEG8/PF4
24 SEG9/PF5
25 SEG10/PF6
26 SEG11/PF7
27 SEG12/PE0
28 SEG13/PE1
29 SEG14/PE2
30 SEG15/PE3
31 SEG16/PE4
32 SEG17/PE5
33 SEG18/PE6
34 SEG19/PE7
35 COM7/SEG20/PD0
36 COM6/SEG21/PD1
37 COM5/SEG22/PD2
38 COM4/SEG23/PD3
39 COM3/PD4
40 COM2/PD5
41 COM1/PD6
42 COM0/PD7
43 PWM0N/INT1/PB0
44 PWM0P/INT2/PB1
45 CLKO/INT3/PB2
46 T1OUT/INT4/PB3
47 TK0/AD0/PB4
48 TK1/AD1/PB5

Pin Summary

Pin number		Pin Name	Type	Input		Output			Function after reset	Alternate Function				Misc
LQFP48	SSOP48			Weak Pull-up	Ext. Interrupt	O.D.	P.O.D.	P.P.		LCD	Touch-Key	UART	SPI	
1	7	TK2/AD2/INT5/PB6	I/O	○	○	○		○	PB6		TK2			
2	8	CAPT/AD3/INT6/PB7	I/O	○	○	○		○	PB7					Capture
3	9	TK3/AD4/INT7/PA1	I/O	○	○		○	○	PA1		TK3			
4	10	TK4/AD5/INT0/PA0	I/O	○	○		○	○	PA0		TK4			
5	11	VCC	P											
6	12	VSS	P											
7	13	FXO/PA3	I/O	○		○		○	OSC out					SYN-CLKO (1)
8	14	FXRC/FXI/PA4	I/O	○		○		○	OSC in					
9	15	SXRC/SXI/PA5	I/O	○		○		○	SOSC in					
10	16	SXO/PA6	I/O	○		○		○	SOSC out					
11	17	VPP/RSTN	I	○					RSTN					
12	18	TK5/T0CKI/PA2	I/O	○			○	○	PA2		TK5			T0CKI
13	19	TK6/SDI/PG0	I/O	○			○	○	PG0		TK6		SDI	
14	20	TK7/SDO/PG1	I/O	○			○	○	PG1		TK7		SDO	
15	21	TK8/SCK/SEG0/PG2	I/O	○		○		○	PG2	S0	TK8		SCK	
16	22	TK9/NSS/SEG1/PG3	I/O	○		○		○	PG3	S1	TK9		NSS	
17	23	TK10/TXD/SEG2/PG4	I/O	○		○		○	PG4	S2	TK10	TXD		
18	24	TK11/RXD/SEG3/PG5	I/O	○		○		○	PG5	S3	TK11	RXD		
19	25	TK12/SEG4/PF0	I/O	○		○		○	PF0	S4	TK12			
20	26	TK13/SEG5/PF1	I/O	○		○		○	PF1	S5	TK13			
21	27	TK14/SEG6/PF2	I/O	○		○		○	PF2	S6	TK14			
22	28	TK15/SEG7/PF3	I/O	○		○		○	PF3	S7	TK15			
23	29	SEG8/PF4	I/O	○		○		○	PF4	S8				
24	30	SEG9/PF5	I/O	○		○		○	PF5	S9				
25	31	SEG10/PF6	I/O	○		○		○	PF6	S10				
26	32	SEG11/PF7	I/O	○		○		○	PF7	S11				
27	33	SEG12/PE0	I/O	○		○		○	PE0	S12				
28	34	SEG13/PE1	I/O	○		○		○	PE1	S13				
29	35	SEG14/PE2	I/O	○		○		○	PE2	S14				

Pin number		Pin Name	Type	Input		Output			Function after reset	Alternate Function				Misc
LQFP48	SSOP48			Weak Pull-up	Ext. Interrupt	O.D.	P.O.D.	P.P.		LCD	Touch-Key	UART	SPI	
30	36	SEG15/PE3	I/O	○		○		○	PE3	S15				
31	37	SEG16/PE4	I/O	○		○		○	PE4	S16				
32	38	SEG17/PE5	I/O	○		○		○	PE5	S17				
33	39	SEG18/PE6	I/O	○		○		○	PE6	S18				
34	40	SEG19/PE7	I/O	○		○		○	PE7	S19				
35	41	COM7/SEG20/PD0	I/O	○		○		○	PD0	C7/S20				
36	42	COM6/SEG21/PD1	I/O	○		○		○	PD1	C6/S21				
37	43	COM5/SEG22/PD2	I/O	○		○		○	PD2	C5/S22				
38	44	COM4/SEG23/PD3	I/O	○		○		○	PD3	C4/S23				
39	45	COM3/PD4	I/O	○		○		○	PD4	C3				
40	46	COM2/PD5	I/O	○		○		○	PD5	C2				
41	47	COM1/PD6	I/O	○		○		○	PD6	C1				
42	48	COM0/PD7	I/O	○		○		○	PD7	C0				
43	1	PWM0N/INT1/PB0	I/O	○	○	○		○	PB0					PWM0N
44	2	PWM0P/INT2/PB1	I/O	○	○	○		○	PB1					PWM0P
45	3	PWM1/CLKO/INT3/PB2	I/O	○	○	○		○	PB2					PWM1
46	4	T1OUT/INT4/PB3	I/O	○	○	○		○	PB3					T1OUT
47	5	TK0/AD0/PB4	I/O	○		○		○	PB4		TK0			
48	6	TK1/AD1/PB5	I/O	○		○		○	PB5		TK1			

Symbol：O.D. = Open Drain
P.O.D. = Pseudo Open Drain
P.P. = Push-Pull Output

(1) Only not Fast XTAL mode can output SYNCLKO (Instruction Cycle).

PIN DESCRIPTION

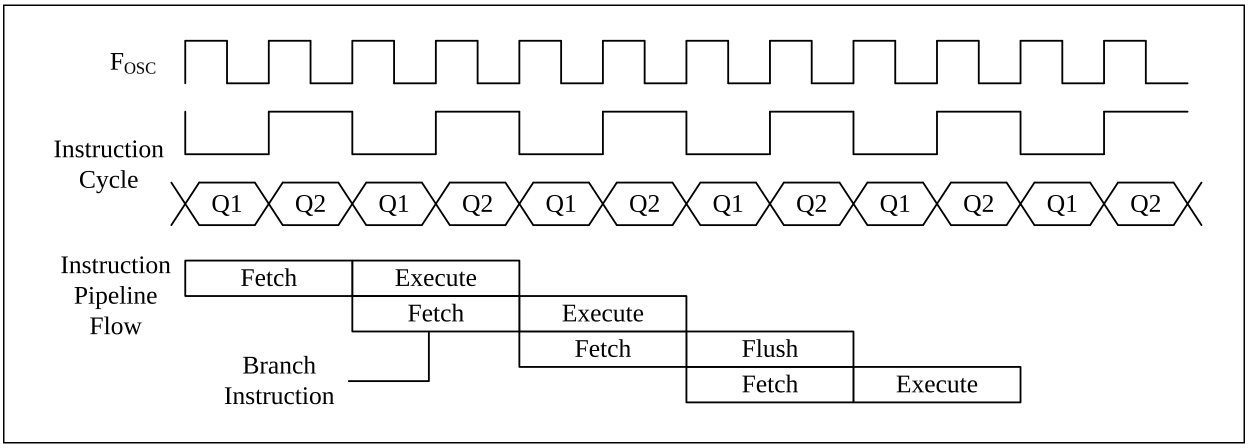
Name	In/Out	Pin Description
PA2–PA0 PG1–PG0	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or "**pseudo-open-drain**" output. Pull-up resistors are assignable by software.
PA6-PA3 PB7-PB0 PD7-PD0 PE7-PE0 PF7-PF0 PG5-PG2	I/O	Bit-programmable I/O port for Schmitt-trigger input, CMOS push-pull output or **open-drain** output. Pull-up resistors are assignable by software.
VPP/RSTN	I	External active low reset
FXI, FXO	-	Fast Crystal/Resonator oscillator connection for system clock
SXI, SXO	-	Slow Crystal/Resonator oscillator connection for system clock
FXRC, SXRC	-	External Fast/Slow RC oscillator connection for system clock
VCC, VSS	P	Power input pin and ground
INT0~INT7	I	External interrupt input
AD0~AD5	I	Analog-to-Digital converter input
TK0~TK15	I	Touch Key input
COM0~COM7 SEG0~SEG23	O	LCD common and segment output
T0CKI	I	Timer0's input in counter mode
CAPT	I	Timer0/Timer1 Capture input
PWM0P, PWM0N	O	PWM0 positive and negative outputs (Period/Duty adjustable)
PWM1	O	PWM1 output (fixed period, duty adjustable)
CLKO	O	System clock output
T1OUT	O	Timer1 match output, T1OUT toggles when Timer1 overflow occurs.
TXD	O	UART data output
RXD	I	UART data input
SDI	I	SPI data input
SDO	O	SPI data output
SCK	I/O	SPI clock output (master) / input (slave)
NSS	I/O	SPI slave select output (master) / input (slave)

FUNCTION DESCRIPTION

1. CPU Core

1.1 Clock Scheme and Instruction Cycle

The system clock is internally divided by two to generate Q1 state and Q2 state for each instruction cycle. The Programming Counter (PC) is updated at Q1 and the instruction is fetched from program ROM and latched into the instruction register in Q2. It is then decoded and executed during the following Q1-Q2 cycle. Branch instructions take two cycles since the fetch instruction is 'flushed' from the pipeline, while the new instruction is being fetched and then executed.



1.2 Addressing Mode

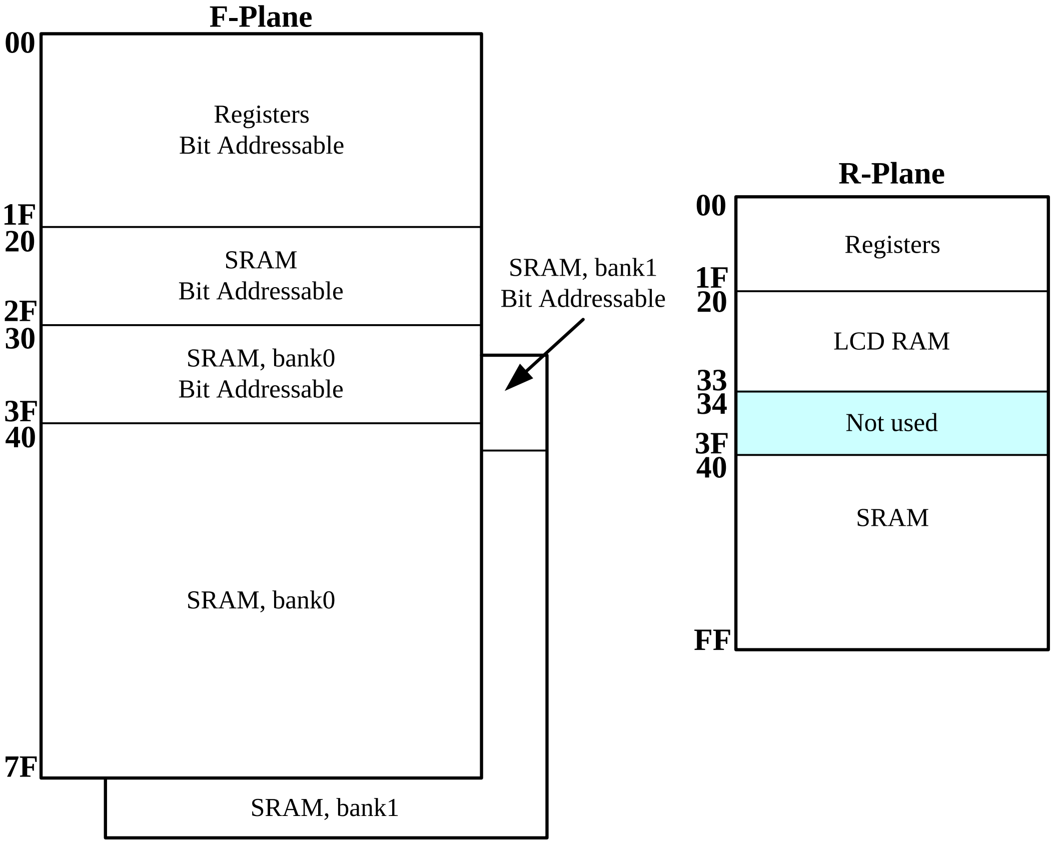
There are two Data Memory Planes in CPU, R-Plane and F-Plane. The F-Plane supports rich instructions operation, such as ADDWF, INCF, MOVWF,..., while the R-Plane only supports MOVWR and MOVRW instructions to exchange data between R-Plane and W-Register.

The lower locations of R-Plane are reserved for the read only registers. Above the registers are the LCD RAM and static RAM. R-plane can be indirect accessed via RSR register (F-plane 07h) and INDR (R-plane 00h).The INDR register is not a physical register. Addressing INDR actually addresses the register whose address is contained in the RSR register (RSR is a pointer).

The lower locations of F-Plane are reserved for the SFR. Above the SFR is General Purpose Data Memory, implemented as static RAM. F-Plane can be addressed directly or indirectly. Indirect Addressing is made by INDF register. The INDF register is not a physical register. Addressing INDF actually addresses the register whose address is contained in the FSR register (FSR is a pointer). The first half of F-Plane is bit-addressable, while the second half of F-Plane is not bit-addressable.

Example:

```
Initial value: [F30h]=12h
MOVLW  30h      ;W=30h
MOVWF  FSR
MOVFW  INDF     ;W=12h
CLRW
MOVWF  INDF     ;[F30h]=00h
INCF   FSR,F    ;FSR=31h
```



1.3 Programming Counter (PC) and Stack

The Programming Counter is 13-bit wide capable of addressing an 8K x 14 program ROM. As a program instruction is executed, the PC will contain the address of the next program instruction to be executed. The PC value is normally increased by one except the followings. The Reset Vector (000h) and the Interrupt Vectors (from 001h to 009h) are provided for PC initialization and Interrupts. For CALL/GOTO instructions, PC loads the lower 12 bits address from instruction word and MSB from STATUS's bit 7. For RET/RETI/RETLW instructions, PC retrieves its content from the top level STACK. For the other instructions updating PC [7:0], the PC [12:8] keeps unchanged. The STACK is 13-bit wide and 8-level in depth. The CALL instruction and Hardware interrupt will push STACK level in order. While the RET/RETI/RETLW instruction pops the STACK level in order.

Since the ROM size is 8K words, it means there are 13 address lines. The CALL/GOTO instructions can load 12 bits address from instruction, that means only 4K size can reach, i.e. either 000h to FFFh; or 1000h to 1FFFh. One ROM page is 4K words in length, so if user needs to CALL/GOTO the other page, the ROM page bit (STATUS.7) must be set/cleared according to page0 or page1 will the program counter be.

Remember that ISR entry addresses are located at ROM Page0, if the user code is interrupted from ROM Page1, ROM Page bit should be cleared when CALL/GOTO will be used in Interrupt Service Routines. While exiting from ISR, user should recall the originally ROM page bit and store to STATUS.7.

1.4 ALU and Working (W) Register

The ALU is 8-bit wide and capable of addition, subtraction, shift and logical operations. In two-operand instructions, typically one operand is the W register, which is an 8-bit non-addressable register used for ALU operations. The other operand is either a file register or an immediate constant. In single operand instructions, the operand is either W register or a file register. Depending on the instruction executed, the ALU may affect the values of Carry (C), Digit Carry (DC), and Zero (Z) Flags in the STATUS register. The C and DC flags operate as a /Borrow and /Digit Borrow, respectively, in subtraction.

Note: /Borrow represents inverted of Borrow register.

/Digit Borrow represents inverted of Digit Borrow register.

The W register can be automatically stored into the internal memory when interrupt and recall when exit from interrupt. This functionality is optional and can be enabled or disabled via ATOSAVE (R-plane CLKCTRL.4) bit.

1.5 STATUS Register

This register contains the arithmetic status of ALU and the Reset status. The STATUS register can be the destination for any instruction, as with any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. It is recommended, therefore, that only BCF, BSF and MOVWF instructions are used to alter the STATUS Register because these instructions do not affect those bits.

PD bit is ‘1’ when SLEEP instruction is executed. It can be cleared by power off-on to generate Power-On Reset, executing CLRWDT, and RSTN goes low.

TO bit is ‘1’ when WDT Timeout is happened. It can be cleared if SLEEP, power off-on, or CLRWDT is executed.

The STATUS register can be automatically stored into the internal memory when interrupt and be restored when exit from interrupt. This functionality is optional and can be enabled or disabled via ATOSAVE (R-plane CLKCTRL.4) bit.

STATUS	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Reset Value	**0**	**0**	**0**	**–**	**–**	**0**	**0**	**0**
R/W	**R/W**	**R/W**	**R/W**	**R**	**R**	**R/W**	**R/W**	**R/W**

Bit	Description	
7	**ROMPAGE**: ROM Page bit 0: ROM Page 0 (address from 000 to FFF) 1: ROM Page 1 (address from 1000 to 1FFF)	
6	**GBIT**: General Purpose Bit No special function. User can use it as general purpose bit.	
5	**RAMBK**: RAM Bank 0: RAM Bank 0 1: RAM Bank 1	
4	**TO**: Time Out 0: after Power-on reset, LVR reset, RSTN or CLRWDT/SLEEP instruction 1: WDT time out occurs	
3	**PD**: Power Down 0: after Power-on reset, LVR reset, RSTN, or CLRWDT instruction 1: after SLEEP instruction	
2	**Z**: Zero Flag 0: the result of a logic operation is not zero 1: the result of a logic operation is zero	
1	**DC**: Decimal Carry Flag or Decimal/Borrow Flag	
	ADD instruction	SUB instruction
	1: a carry from the low nibble bits of the result occurs 0: no carry	1: no borrow 0: a borrow from the low nibble bits of the result occurs
0	**C**: Carry Flag or Borrow Flag	
	ADD instruction	SUB instruction
	1: a carry occurs from the MSB 0: no carry	1: no borrow 0: a borrow occurs from the MSB

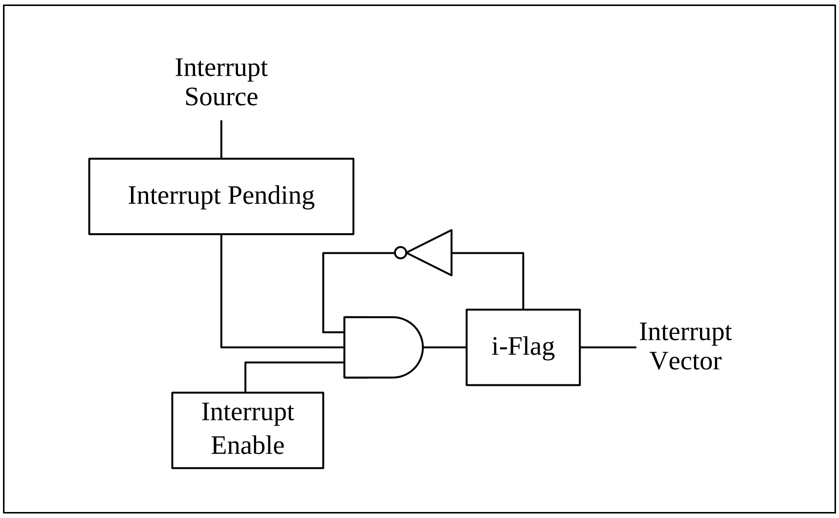
1.6 Interrupt

The TM57FLA80 has 1 level, 9 vectors and 15 interrupt sources. Each interrupt source has its own enable control bit. An interrupt event will set its individual interrupt flag, no matter its interrupt enable control bit is 0 or 1. Because TM57FLA80 has 9 vectors, there is not an interrupt priority register. Priority of each interrupt is different from each other and the device does not support nested interrupt. Another interrupt can be executed only if the current interrupt is exited (that is, RETI instruction is executed). Although the interrupts do not have priority, however, when exiting from current interrupt, if there are more than 2 interrupt happen, the priority of the interrupt is:

UART > SPI > TM0 > TM1 > TM2 > XINTA > XINTB > WKT > PWM0

Priority	Address	Source	Description	WakeUp
3	001	Timer0	Timer0 Counter Overflow	
4	002	Timer1	Timer1 Counter Overflow	
5	003	Timer2	Timer2 Count Match	Yes
9	004	PWM0	PWM0 Period Finish	
8	005	WKT	Wakeup Timer Match (if WDT disable)	Yes
6	006	XINTA	PA0, PB0, PB1 falling interrupt (PA0 is rising/falling selectable)	Yes
7	007	XINTB	PA1, PB2, PB3, PB6, PB7 falling interrupt (PB7 is rising/falling selectable)	Yes
1	008	UART	UART TX/RX Complete	
2	009	SPI	SPI TX/RX Complete	

If the corresponding interrupt enable bit has been set (INTE), it would trigger CPU to service the interrupt. CPU accepts interrupt in the end of current executed instruction cycle. In the mean while, a “CALL 00n” (n ranges from 1 to 9) instruction is inserted to CPU, and i-flag is set to prevent recursive interrupt nesting. The i-flag is cleared in the instruction after the “RETI” instruction. That is, at least one instruction in main program is executed before service the pending interrupt. The interrupt event is level triggered. F/W must clear the interrupt event register while serving the interrupt routine.



2. Chip Operation Mode

2.1 Reset

The TM57FLA80 can be RESET in four ways.

- Power-On-Reset
- Low Voltage Reset (LVR)
- External Pin Reset (RSTN)
- Watchdog Reset (WDT)

After Power-On-Reset, all system and peripheral control registers are then set to their default hardware Reset values. The clock source, LVR level and chip operation mode are selected by the SYSCFG register value.

The Low Voltage Reset features static reset when supply voltage is below a threshold level. There are two threshold levels can be selected. The LVR's operation mode is defined by the SYSCFG register.

There are two voltage selections for the LVR threshold level, one is higher level which is suitable for application with Vcc is more than 3.3V, while another one is suitable for application with Vcc is less than 3.3V. See the following LVR Selection Table; user must also consider the lowest operating voltage of operating frequency.

LVR Selection Table:

LVR Threshold Level	Consider the operating voltage to choose LVR
LVR2.9	5.5V > Vcc > 3.3V
LVR2.1	Vcc is wide voltage range

The External Pin Reset and Watchdog Reset can be disabled or enabled by the SYSCFG register. These two resets also set all the control registers to their default reset value. The TO/PD flag is not affected by these resets.

2.2 System Configuration Register (SYSCFG)

The System Configuration Register (SYSCFG) is located at Flash INFO area. The SYSCFG determines the option for initial condition of MCU. It is written by FLASH Writer only. User can select clock source, LVR threshold voltage and chip operation mode by SYSCFG register. The 13th bit of SYSCFG is code protection selection bit. If this bit is 1, the data in FLASH ROM will be protected, when user reads FLASH ROM.

Bit	**13~0**	
Default Value	**00_0000_0000_0000**	
Bit	Description	
13	**PROTECT**: Code Protection Selection	
	1	Code protection
	0	No protect
12	**IVCPD**: IVC*/LVR Power Down in Stop mode	
	1	IVC/LVR OFF in Stop mode
	0	IVC/LVR ON in Stop mode
11	**LVR**: LV reset mode	
	1	LVR threshold is 2.1V
	0	LVR threshold is 2.9V
10	1	LVR enable
	0	LVR disable
9-8	**CLKS**: Fast Clock Source Selection	
	11	Fast Xtal (1 MHz~12 MHz)
	01	FIRC (4 MHz)
	00	External RC
		Slow Clock is register controlled
7	**3V/5V Selection** (TM57FLA80 only, this bit DOESN’T support TM57FLA80A)	
	1	Vcc maximum working voltage at 3.3V, slow mode save about 80 uA when Vcc =3V
	0	Vcc maximum working voltage at 5.5V, slow mode will consume about 80 uA when Vcc =3V
6	**WDTE**: WDT Reset Enable	
	1	Enable WDT Reset, Disable WKT Timer
	0	Disable WDT Reset, Enable WKT Timer
5	Not used	
4-0	**IRCF**: FIRC frequency adjustment control	

* IVC is the chip built-in voltage regulator for internal circuit. IVC can be powered down when Stop or Idle mode, and IVC_REG should be set to the corresponding working Vcc supplied voltage.

* LVR is the Low-voltage Reset circuit.

2.3 Flash ROM

The FLASH Program ROM of this device is 8K words, with an extra INFO area to store the SYSCFG and manufacture data. The FLASH ROM can be written multi-times and can be read as long as the PROTECT bit of SYSCFG is not set. The SYSCFG can be read no matter PROTECT is set or cleared, but can be written only when PROTECT is not set or FLASH ROM is blank. That is, unprotect the PROTECT bit can be done only if the Program ROM area is blank. The tenx certified writer can do the above actions with the sophisticated software.

Program Memory

0000	Reset Vector
0001	TMR0 interrupt
0002	TMR1 interrupt
0003	TMR2 interrupt
0004	PWM0 interrupt
0005	WKT interrupt
0006	XINTA interrupt
0007	XINTB interrupt
0008	UART interrupt
0009	SPI interrupt
1fff	

Config Memory

00	SYSCFG
01	Manufacture Reserved Area
1F	

2.4 System Clock and Operation Mode Selection

TM57FLA80 is designed with multi-clock system. There are five kinds of clock source, FAST XTAL Clock, FAST XRC Clock, Slow XTAL Clock, Slow XRC Clock, and FIRC. Each clock source can be applied to CPU kernel as system clock. When in Idle mode, only Slow Clock can be configured to keep oscillating to provide clock source to LCD block and Timer2 block. Refer to the Figure 2.4.1.

TM57FLA80 is operated in one of four modes: Fast Mode, Slow Mode, Idle mode, and Stop mode.

2.4.1 Fast Mode

In Fast Mode, TM57FLA80 can select Fast XTAL, Fast XRC or FIRC as its CPU clock by SYSCFG bit9 and bit8 setting. Besides, firmware can also enable the Slow Clock for the Timer2 and LCD system operating.

In this mode, the program is executed using Fast Clock as CPU clock. The Timer0, Timer1, ADC, PWM0, PWM1, UART, and SPI blocks are also driven by Fast Clock. Timer2 can also be driven by Fast Clock by setting TIMER2CLK to “1” and SELSUB to “0”.

2.4.2 Slow Mode

In Slow Mode, TM57FLA80 can select Slow XTAL or Slow XRC as its CPU clock by R-plane control register (CLKCTRL). In this mode, the Fast Clock is stopped and Slow Clock is enabled for power saving. All peripheral blocks such as Timer0, Timer1, Timer2, ADC, PWM0, PWM1, UART, and SPI are driven by Slow Clock in the Slow Mode.

2.4.3 Idle Mode

If Slow Clock is enabled before executing the SLEEP instruction, the TM57FLA80 enters the “Idle Mode”. In this mode, the Slow Clock will continue running to provide clock to LCD and Timer2 block and keep the LCD COM and SEG pins scanning. CPU stop fetching code and all blocks are stop except LCD/Timer2 related circuits.

2.4.4 Stop Mode

If Slow Clock is disabled before executing the SLEEP instruction, every block is turned off and the TM57FLA80 enters the “Stop Mode”. In this mode, the internal IVC and LVR can also be powered down depends on SYSCFG bit12. Stop Mode is similar to Idle Mode. The difference is all clock oscillators either Fast Clock or Slow Clock is powered down and no clock is generated. Only the on-chip Wake-up Timer is still counting for wakeup if the WDTE bit of SYSCFG is “0”.

Watchdog Timer and Wake-up Timer share one physical timer, it means if WDTE is equal to 1, the Wake-up Timer function is disabled. Conversely, if the WDTE is cleared to “0” and WKTIE is set to “1”, the Wake-up Timer is enabled and will consume little power to count when in Stop Mode.

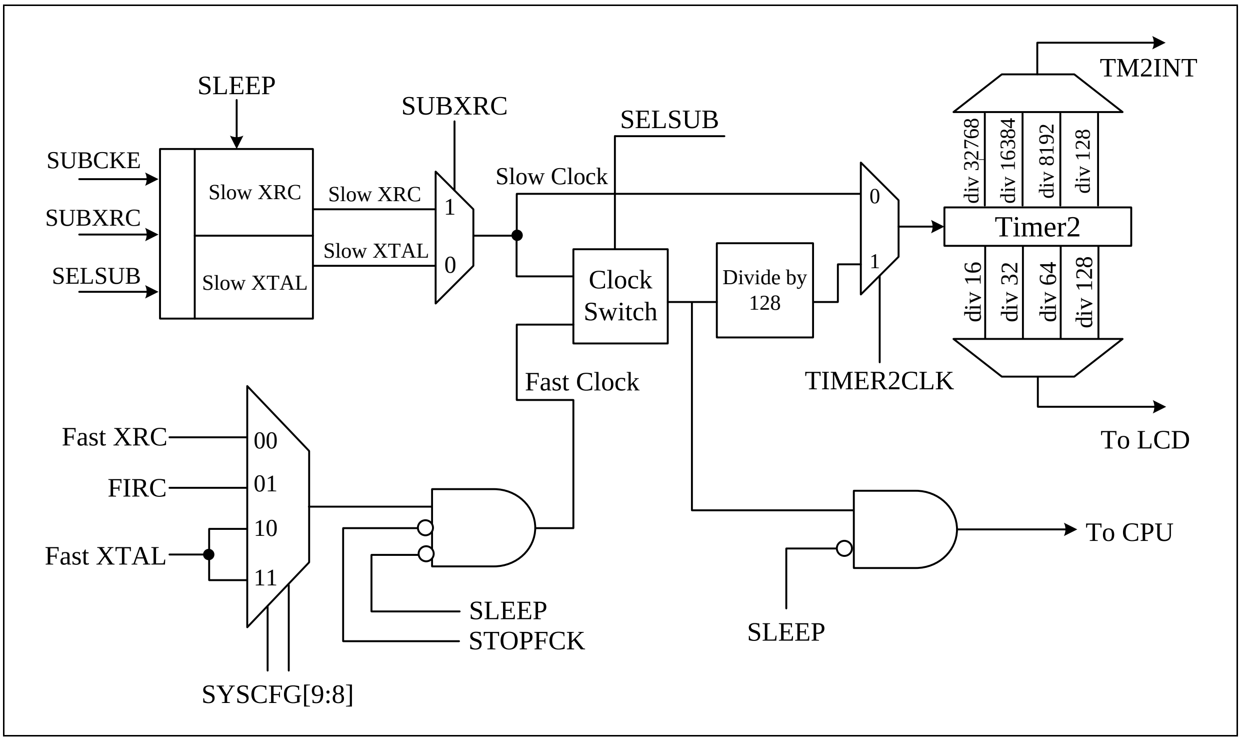
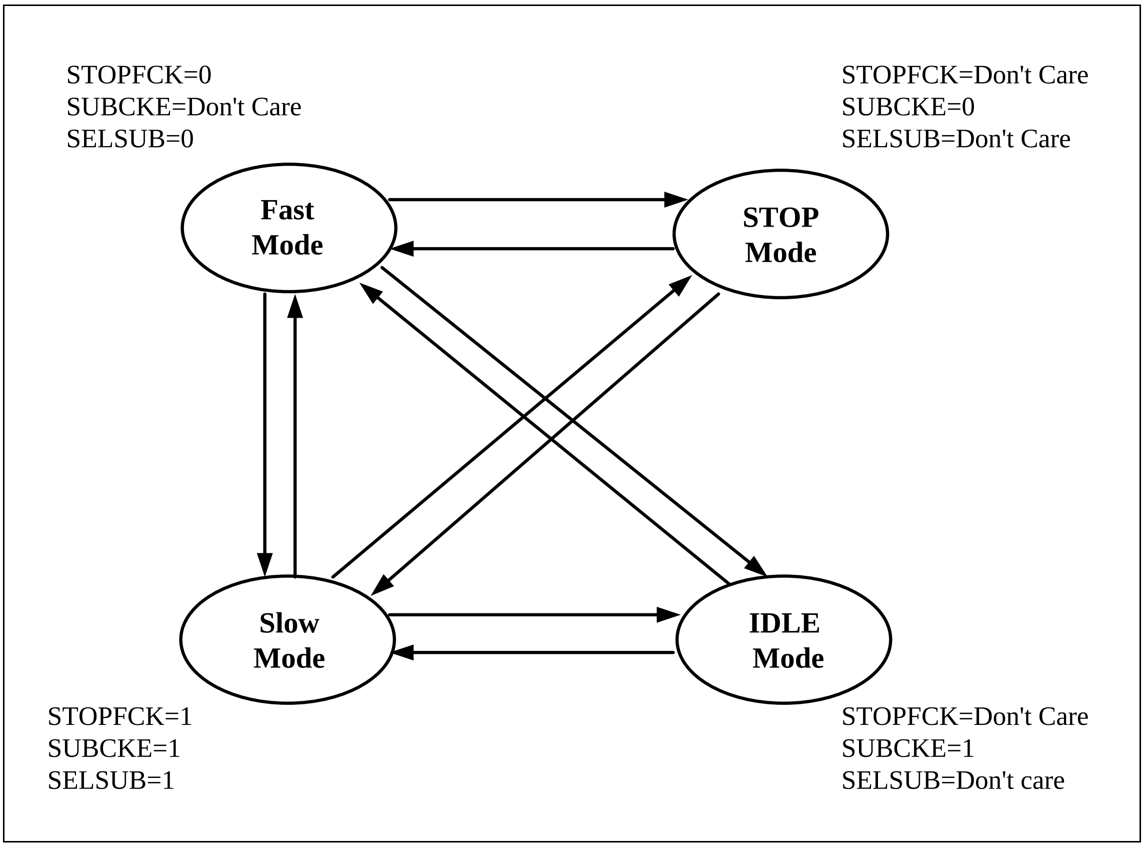


Figure 2.4.1 Clock Scheme Block Diagram

2.5 Modes Transition Diagram



Fast Mode can be chosen by SYSCFG[9:8] when equals to 11 (Fast XTAL) , 00 (Fast XRC) , or 01 (FIRC). The following steps are suggested to be executed by order when Fast Mode transits to Slow Mode:

1. Enable Slow Clock (SUBCKE=1)
2. Switch to Slow Clock (SELSUB=1)
3. Stop Fast clock (STOPFCK=1)

Note that if the SUBCKE=0, the Slow Clock oscillator can also be enabled if SELSUB=1 while not in power-down mode. Once the SLEEP is executed, the Slow Clock oscillator will be turned off immediately and the chip is entering Stop Mode. Neither Fast nor Slow clock is oscillating to achieve power saving.

Slow Mode can be enabled by SUBCKE bit and SELSUB bit in CLKCTRL register. The following steps are suggested to be executed by order when Slow Mode transits to Fast Mode:

1. Enable Fast Clock (STOPFCK =0)
2. Switch to Fast Clock (SELSUB=0)
3. Stop Slow Clock (SUBCKE=0) -------- this is optional. Slow Clock can keep oscillating when in Fast Mode.

Idle Mode means only Slow Clock is oscillating to provide clock to Timer2/LCD block, meanwhile, the CPU stops executing instructions. The Timer2 is used to generate LCD clock is still counting. The internal voltage regulator (IVC) can be disabled by configword IVCPD bit, and the proper IVC_REG must be set depends on Vcc voltage ranges. The Idle Mode can be configured by following setting in order:

1. SUBCKE=1
2. SLEEP

Idle Mode can be woken up by XINTA, XINTB, Wake-up Timer, and Timer2 interrupt.

Stop Mode can be entered by executing SLEEP instruction while SUBCKE=0. Stop Mode can be woken up by XINTA, XINTB, and Wake-up Timer.

3. Peripheral Functional Block

3.1 Watchdog (WDT) / Wakeup Timer (WKT)

The WDT and WKT share the same timer which is clocked by on-chip oscillator. The overflow period of WDT/WKT can be selected from 3.45 ms to 112 ms. The WDT/WKT is cleared by the CLRWDT instruction. If the Watchdog Reset is enabled (WDTE=1), the WDT generates the chip reset signal, otherwise, the WKT only generates overflow time out interrupt. The WDT/WKT works in all 4 kinds of mode. If WDTE=0 and WKTIE=0 (Wakeup interrupt disable), the WDT/WKT stop counting for power saving.

If the WDTE=1 and WKTIE=0, WDT/WKT timer will be cleared and stopped to power saving in Stop Mode. If the WDTE=1 and WKTIE=1, WDT/WKT timer will keep counting in Stop Mode. Refer to the following table and figure.

Mode	WDTE	WKTIE	WDT/WKT *
Normal Mode	0	0	Stop
	0	1	Run
	1	0	
	1	1	
Stop / Idle Mode	0	0	Stop
	0	1	Run
	1	0	Stop
	1	1	Run

If the user program needs the MCU totally shut down for power conservation in Stop mode, the above setting of control bits should be followed.

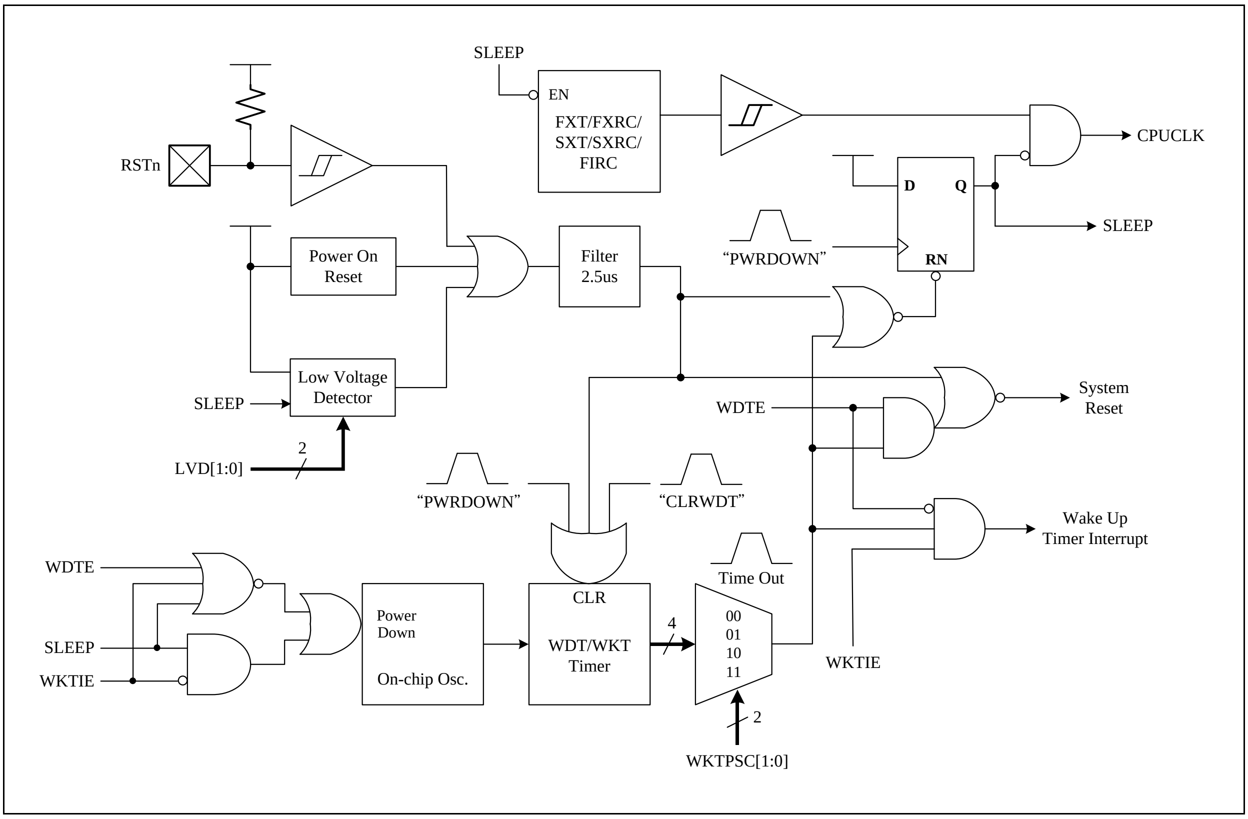


Figure 3.1. The internal Reset scheme

3.2 8-bit Timer/Counter/Capture (Timer0) with Pre-scaler (PSC)

The Timer0 is an 8-bit wide register of F-Plane 01h. It can be read or written as any other registers of F-Plane. Timer0 increases itself periodically and rolls over based on the pre-scaled clock source, which can be instruction cycle, T0CKI (PA2) rising/falling, or Touch Key oscillating clock rising/falling. The Timer0 increasing rate is determined by “Timer0 Prescale” (TM0PSC) register in R-Plane. The Timer0 will generate interrupt when it counts to overflow if Timer0 interrupt Enable (TM0IE) is set.

Timer0 can be stopped counting if the STOPTM0 bit is set. Timer0 can be configured as capture mode. If T0CAPTURE bit is set to “1”, Timer0 will not count until the CAPT pin (i.e. PB7) is active.

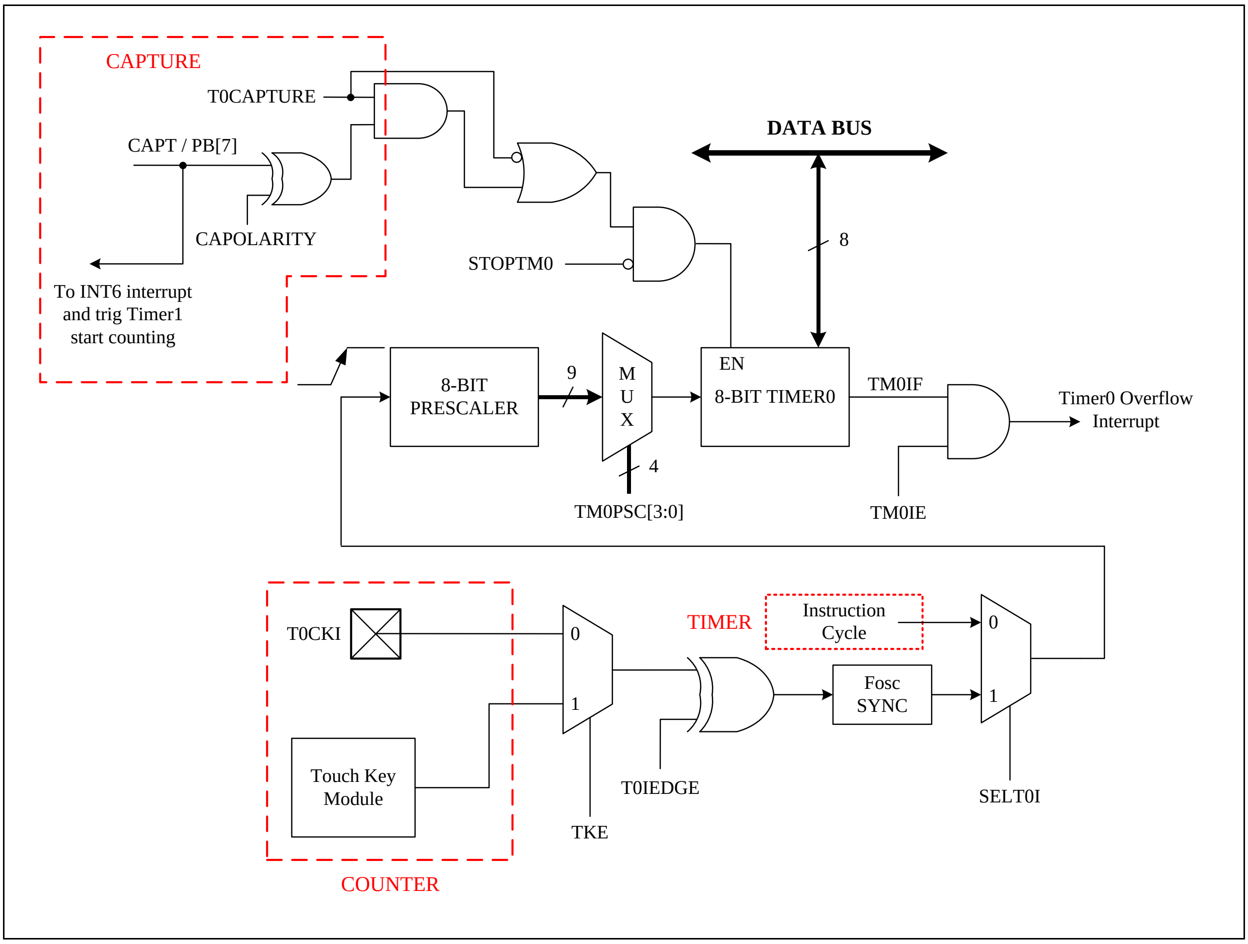


Figure 3.2.1 Timer0 Block Diagram

Figure 3.2.2 shows the Timer0 works in pure timer mode. When the Timer0 prescaler (TM0PSC) is written, the internal 8-bit prescaler will be cleared to 0 to make the counting period correct at the first Timer0 count. TM0CLK is the internal signal that causes the Timer0 to increase by 1 at the end of TM0CLK. TM0WR is also the internal signal that indicates the Timer0 is directly written by instruction, meanwhile, the internal 8-bit prescaler will be cleared. When Timer0 counts from FFh to 00h, TM0IF (Timer0 Interrupt Flag) will be set to 1 and generate interrupt if TM0IE (Timer0 Interrupt Enable) is set.

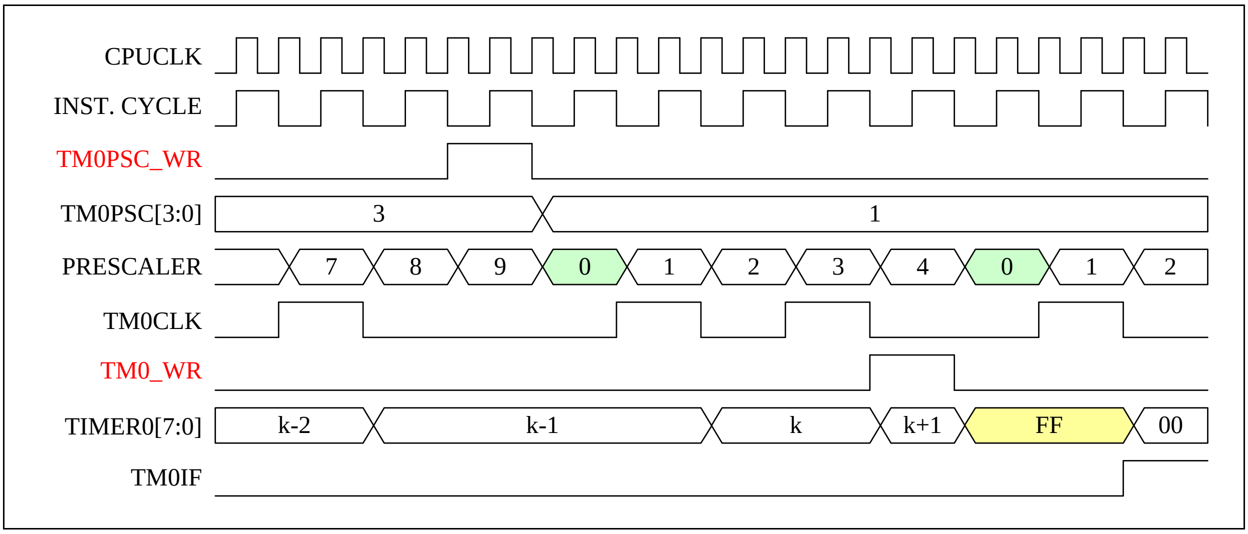


Figure 3.2.2 Timer0 works in timer mode

The following timing diagram describes the Timer0 works in counter mode. If SELT0I=1 then the Timer0 counter source clock is from T0CKI pin or Touch Key module that depends on TKE bit. T0IEDGE bit determines whether rising or falling edge to clock the Timer0 prescaler counter. As shown in Figure 3.2.3, T0CKI (or Touch Key clock) signal is synchronized by instruction cycle (i.e. 2 oscillation clocks), that means the high/low time durations of T0CKI must be longer than one instruction cycle time to guarantee each T0CKI's change will be detected correctly by the synchronizer.

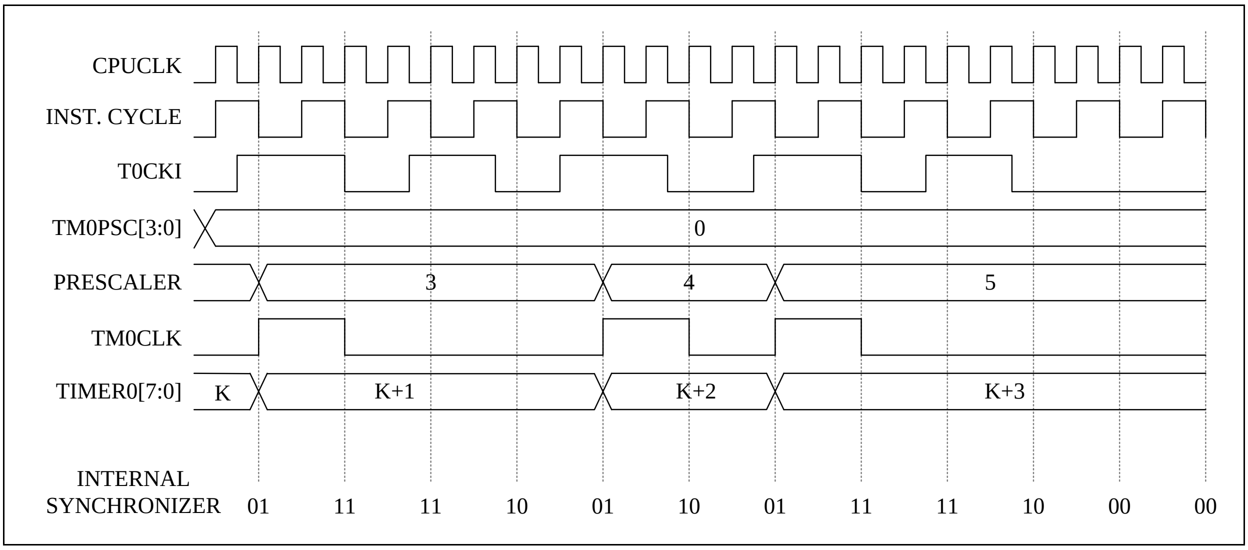


Figure 3.2.3 Timer0 works in external T0CKI input mode

Timer0 can also be used to measure the pulse with and period capture on CAPT pin. This function needs the Timer1 and INT6 external interrupt and software control step by step. Section 3.5 will discuss the details.

3.3 Timer1

Timer1 is a 16-bit counter with 16-bit auto-reload register. Figure 3.3.1 shows the Timer1 block diagram. Timer1 can only be accessed by reading F-plane TM1H and TM1L. Writing TM1H and TM1L is actually writing to Timer1 reload registers. The clock sources of Timer1 are Fosc and Fosc/2, selected by TM1PSC. Setting the bit CLRTM1 will clear Timer1 and hold Timer1 on 0000h. Setting the STOPTM1 bit will stop Timer1 counting. T1OUT is an output signal that toggles when Timer1 overflows.

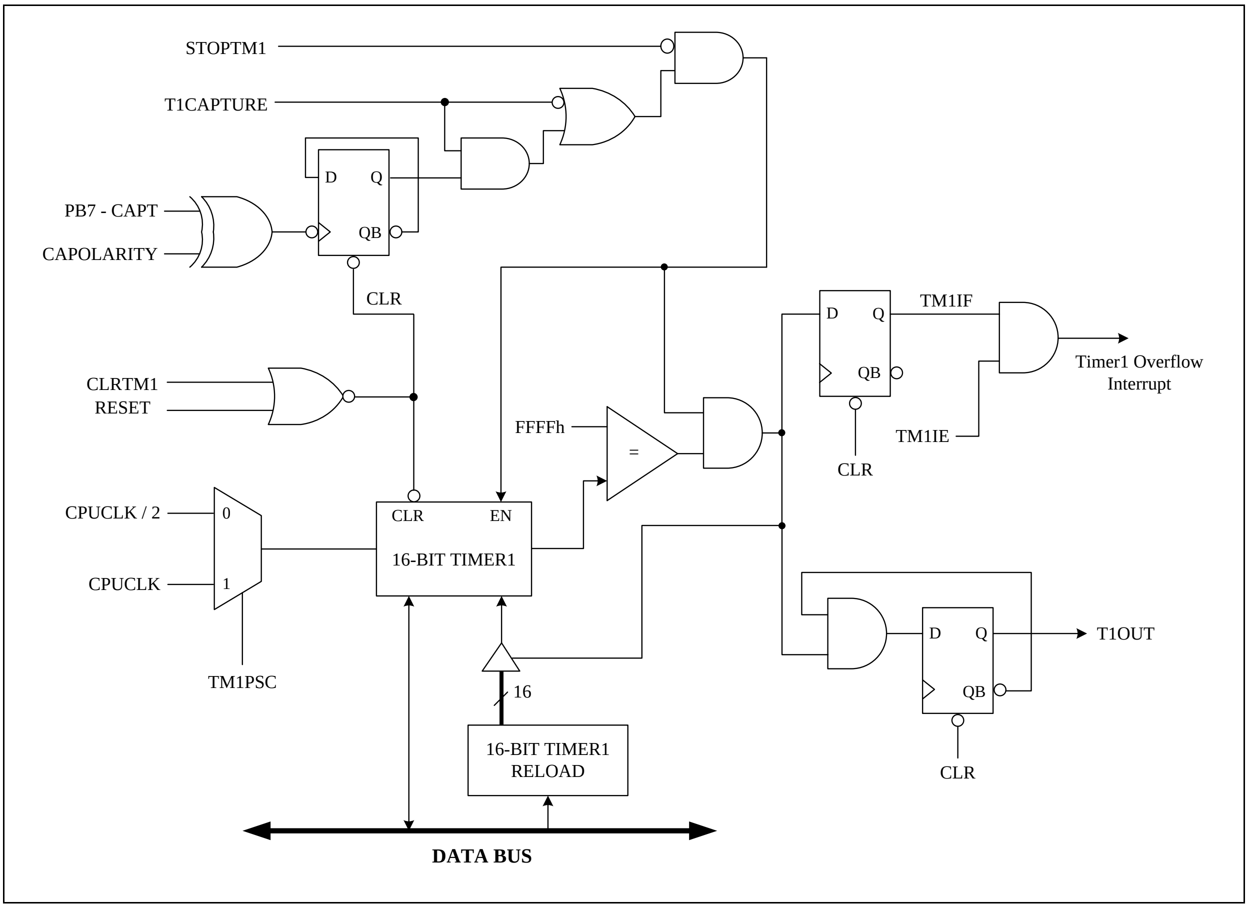


Figure 3.3.1 Timer1 Block Diagram

Note that writing to TM1H and TM1L is actually writing to Timer1 reload register, while reading TM1H and TM1L is actually reading the Timer1 counter itself. That is, Timer1 counter and Timer1 Reload register share two addresses (0ah, 0bh) of F-plane.

Timer1 can also work with capture mode. When works in capture mode, Timer1 will start counting when the CLRTM1 bit is cleared and the first falling edge of CAPT pin (if CAPOLARITY=0) is coming. When the 2nd falling edge of CAPT pin is coming, Timer1 stops counting and holds the value. When the 3rd falling edge of CAPT pin is coming, the Timer1 continues counting. Figure 3.3.2 shows the detail timing diagram.

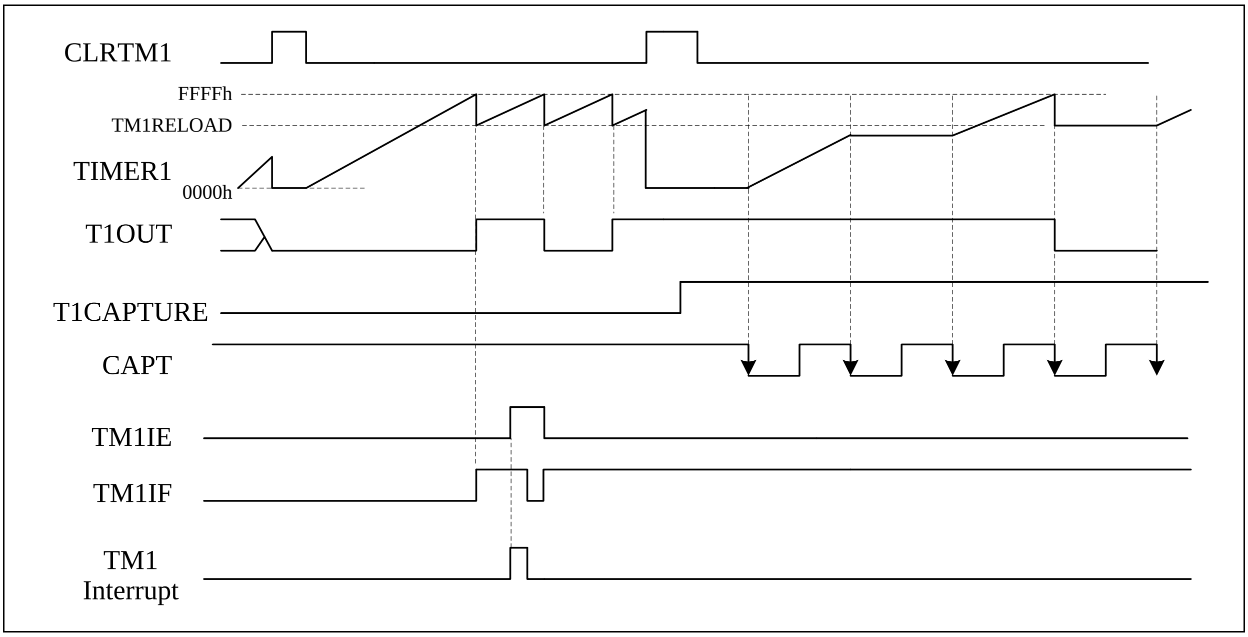


Figure 3.3.2 (CAPOLARITY=0, implies CAPT falling edge)

3.4 Timer2

Timer2 is a 15-bit counter and the clock sources are from either CPUCLK/128 or Slow Clock. It is used to generate time base interrupt and LCD clock. The Timer2 cannot be read by instructions. It generates interrupt by the selected clock divided by 32768, 16384, 8192, and 128, depends on TIMER2DIV register bits. Figure 3.4.1 shows the block diagram of Timer2.

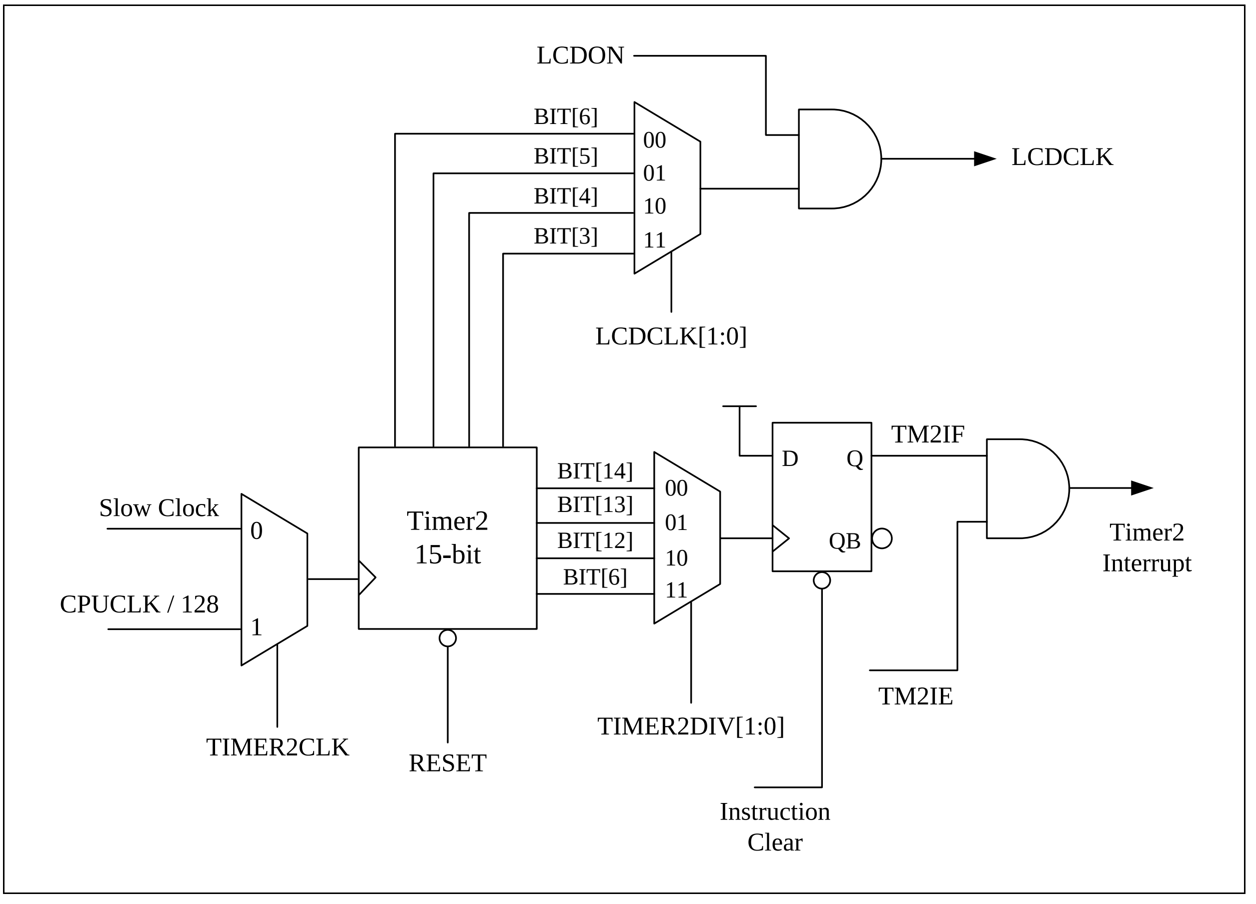


Figure 3.4.1 Timer2 Block Diagram

3.5 Timer0 and Timer1 used for Pulse Width and Period Capture

Timer0 and Timer1 can cooperate to measure the signal period and duty cycle time. The key is multifunction of PB7 (CAPT, INT6). Suppose that:

- SELT0I=0, Timer0 prescaler increases per instruction cycle.
- T0CAPTURE=1, T1CAPTURE=1. Timer0 and Timer1 work in capture mode.
- TKE=0, Touch Key function is disabled.
- INT6EDGE=0, PB7 pin (CAPT pin) interrupt every falling edge.
- CAPOLARITY=0, **Timer1** starts/holds in turn when PB7 pin (CAPT pin) falling edge is coming. **Timer0** starts counting when PB7 pin (CAPT pin) in logic ‘1’ level and hold the Timer0 value when PB7 pin (CAPT pin) is in logic ‘0’ level.
- Timer1 is used to measure the signal period, Timer0 is used to measure the PB7 (CAPT pin) in logic ‘1’ time (i.e. the duty cycle of the signal).

Figure 3.5.1 shows how to use Timer0 and Timer1 to measure the PB7 (CAPT pin) signal’s period and duty cycle.

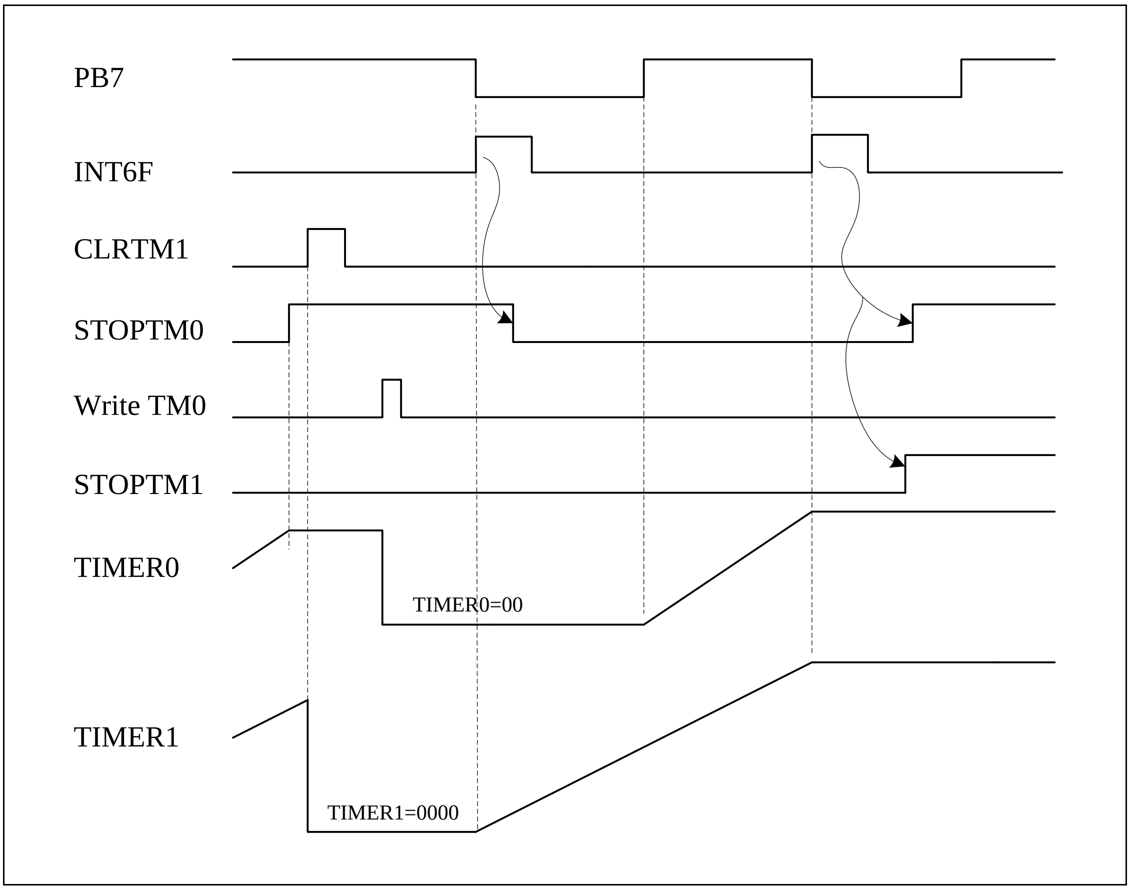


Figure 3.5.1 Timer0 and Timer1 are used to measure the signal on CAPT pin.

Follow the steps below to start measuring the CAPT pin's period and duty cycle.

1. Stop Timer0 by firmware (Timer0 will be stopped and hold).
2. Clear Timer1 by firmware.
3. Clear Timer0 by directly write 00h to Timer0 (Timer0 is still hold).
4. Once PB7 falling edge is coming, the Timer1 starts counting; meanwhile the XINT6 interrupt is generated and clears the STOPTM0 by firmware. Now the Timer0 is ready to count when PB7 goes high.
5. PB7 rising edge is coming, Timer0 starts counting until the PB7 returns to 0 and holds the counting value. Timer1 also stops counting and holds the value.
6. XINT6 interrupt is generated again, firmware stops Timer1 and Timer0 to read the period and duty cycle.

It is not necessary to use both Timer0 and Timer1. If only the duty cycle (CAPT high time) needs to be measured, there is no need to use Timer1 to measure the period. In such case, user can set the T0CAPTURE=1, T1CAPTURE=0, and CAPOLARITY=0 (capture the high time duration). As shown in Figure 3.5.2, Timer0 is counting up only when PB7 (CAPT pin) is '1'. Note that the internal prescaler will be kept to next Timer0 count, so it will not lose the counting accuracy.

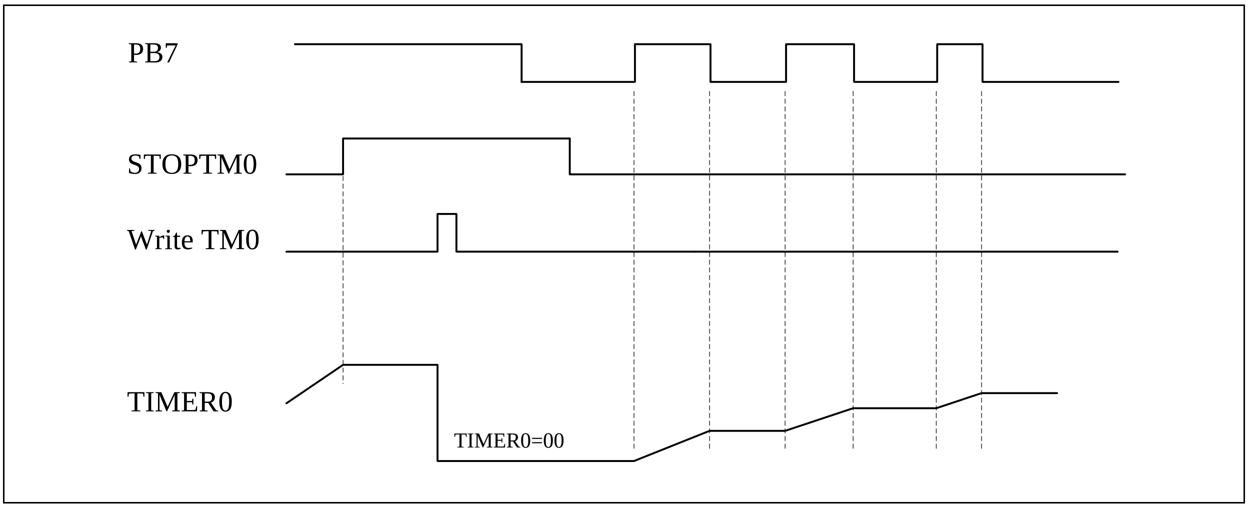


Figure 3.5.2 Timer0 is used to measure the high (or low) time on CAPT pin

3.6 PWM0

The chip has a built-in 8-bit PWM generator. The source clock comes from CPUCLK divided by 1, 2, 4, and 8. The PWM0 duty cycle can be changed with writing to PWM0DUTY, writing to PWM0DUTY will not change the current PWM duty until the current PWM period completes. When current PWM period is finish, the new value of PWM0DUTY will be updated to the PWM0BUF.

The PWM0P will be output to PB1 if PWM0PE is set to 1. The complement of PWM0P, PWM0N, will be output to PB0 if PWM0NE is set to 1. Also, the PWM period complete will generate an interrupt when PWM0IE is set to 1. Setting the CLRPWM0 bit will clear the PWM0 counter and load the PWM0DUTY to PWM0BUF, CLRPWM0 bit must be cleared so that the PWM0 counter can count.

Note that the default value of CLRPWM0 bit is ‘1’.

Figure 3.6.1 shows the block diagram of PWM0.

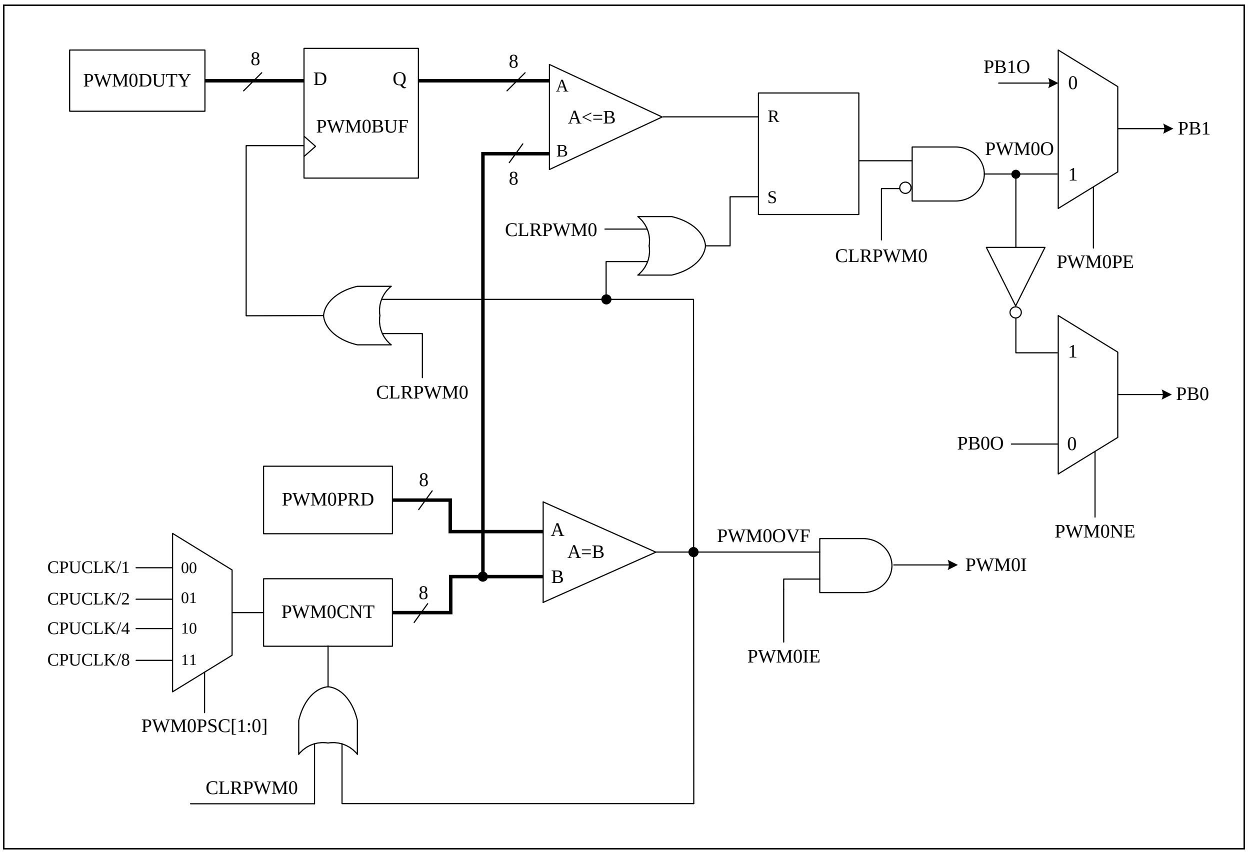


Figure 3.6.1 PWM0 Block Diagram

Figure 3.6.2 and Figure 3.6.3 shows the PWM0 waveforms. When CLRPWM0 bit is set to ‘1’, the PWM0 output is cleared to ‘0’ no matter what its current status is. Once the CLRPWM0 bit is cleared to ‘0’, the PWM0 output is set to ‘1’ to begin a new PWM cycle. PWM0 output will be ‘0’ when PWM0CNT is greater than or equals to PWM0BUF. PWM0CNT keeps counting up when equals to PWM0PRD, the PWM0 output is set to ‘1’ again.

The PWM0 period can be set by writing period value to PWM0PRD (R.07h) register. Note that changing the PWM0PRD immediately changes the PWM0PRD value in the Figure 3.6.1 that is different from PWM0DUTY which has PWM0BUF to update the duty at the end of current period. The programmer must pay attention to the correct time to change PWM0PRD. By observing the Figure 3.6.1, there is a digital comparator that compares the PWM0CNT and PWM0PRD, if PWM0CNT is larger than PWM0PRD after setting the PWM0PRD, a fault long PWM cycle will be generated because PWM0CNT must count to overflow then keep counting to PWM0PRD to finish the cycle.

It is recommended that the programmer should modify the PWM0PRD only when PWM0 interrupt happens and it makes sure the fault long PWM cycle will not be generated.

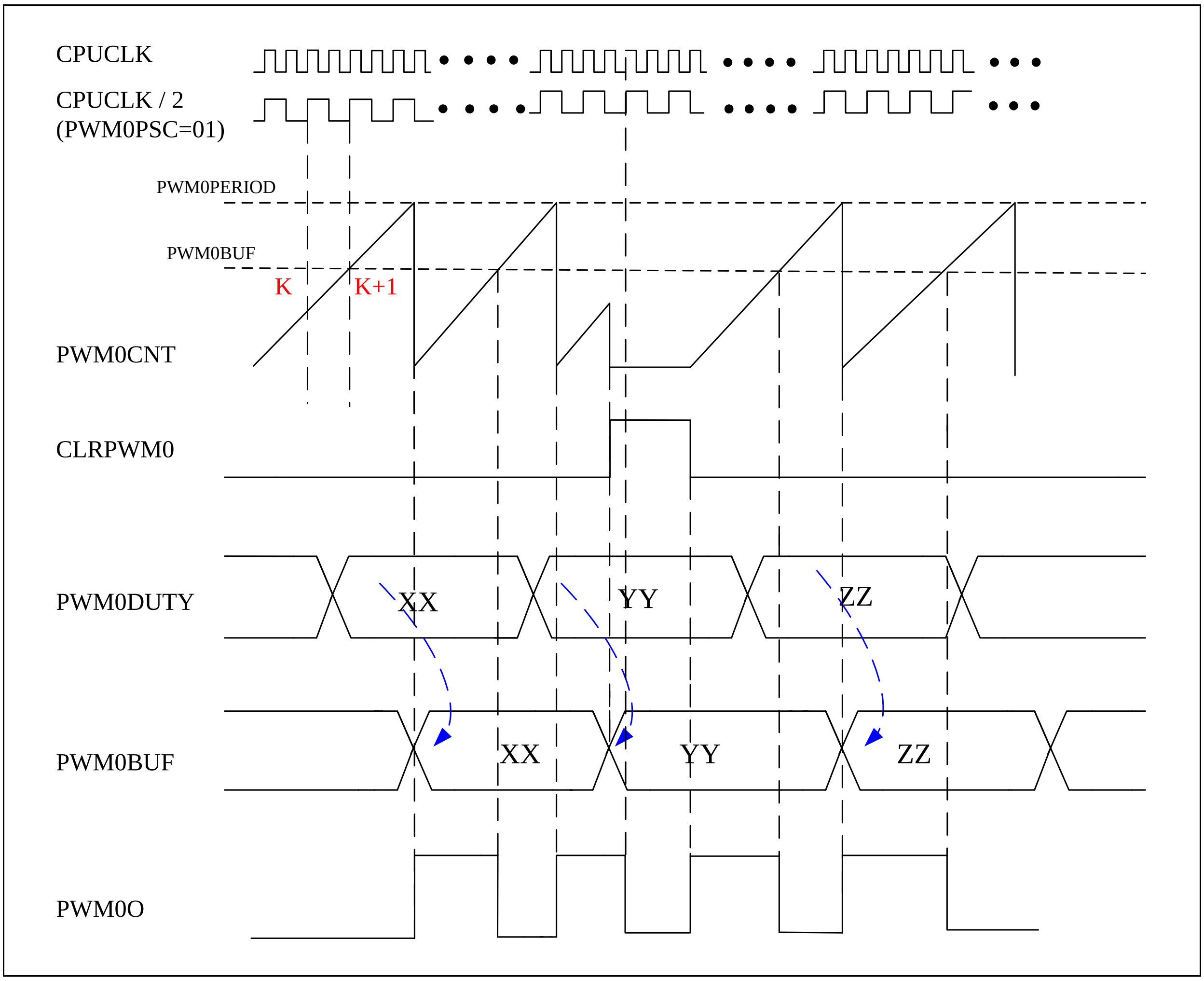


Figure 3.6.2 PWM0 Timing (CLRPWM0 before PWM0CNT reaches PWM0BUF)

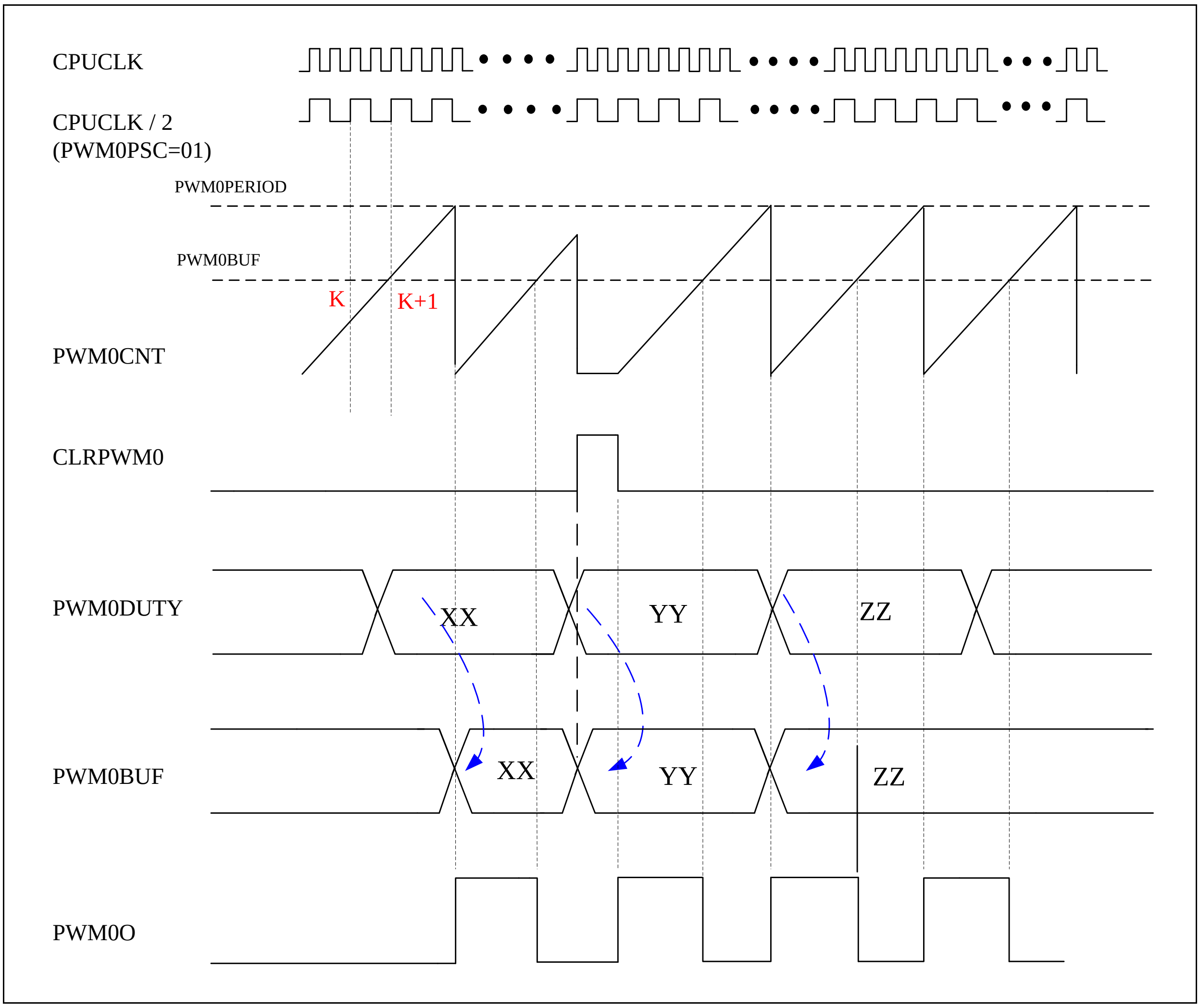


Figure 3.6.3 PWM0 Timing (CLRPWM0 after PWM0CNT over PWM0BUF)

3.7 PWM1

PWM1 is a simple fixed frequency and duty cycle variable PWM generator. The PWM frequency is fixed, the period is system clock counts from 0 to 255. The duty can be set via PWM1DUTY register. The output of PWM1 shares the pin PB2 that can be selected by PWM1E control bit. Figure 3.7.1 is the block diagram of PWM1. Figure 3.7.2 shows the related timing of PWM1. The PWM frequency is:

PWM1 Frequency = CPUCLK / 256

PWM1 Duty Cycle = (PWM1DUTY / 256) * 100%

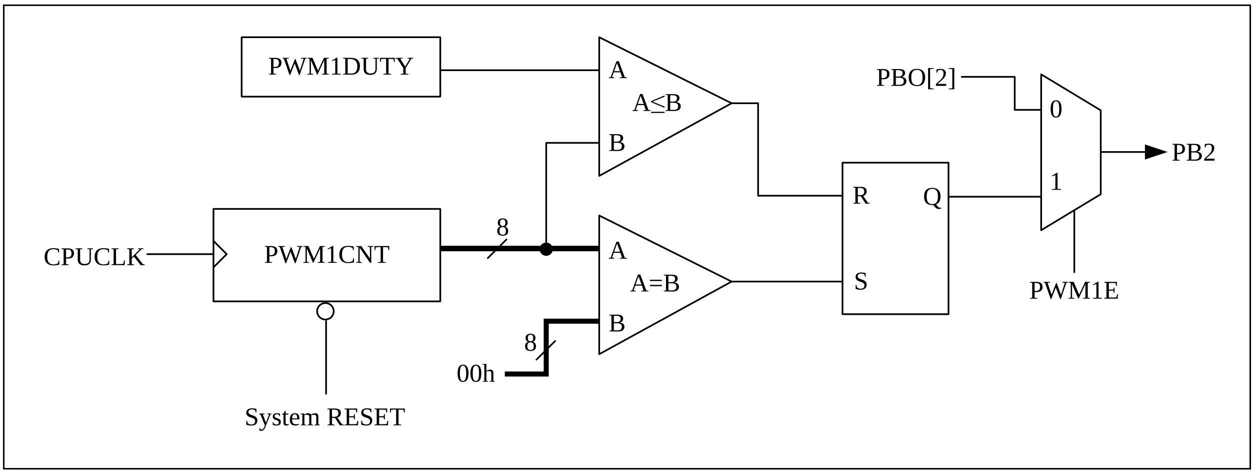


Figure 3.7.1 PWM1 Block Diagram

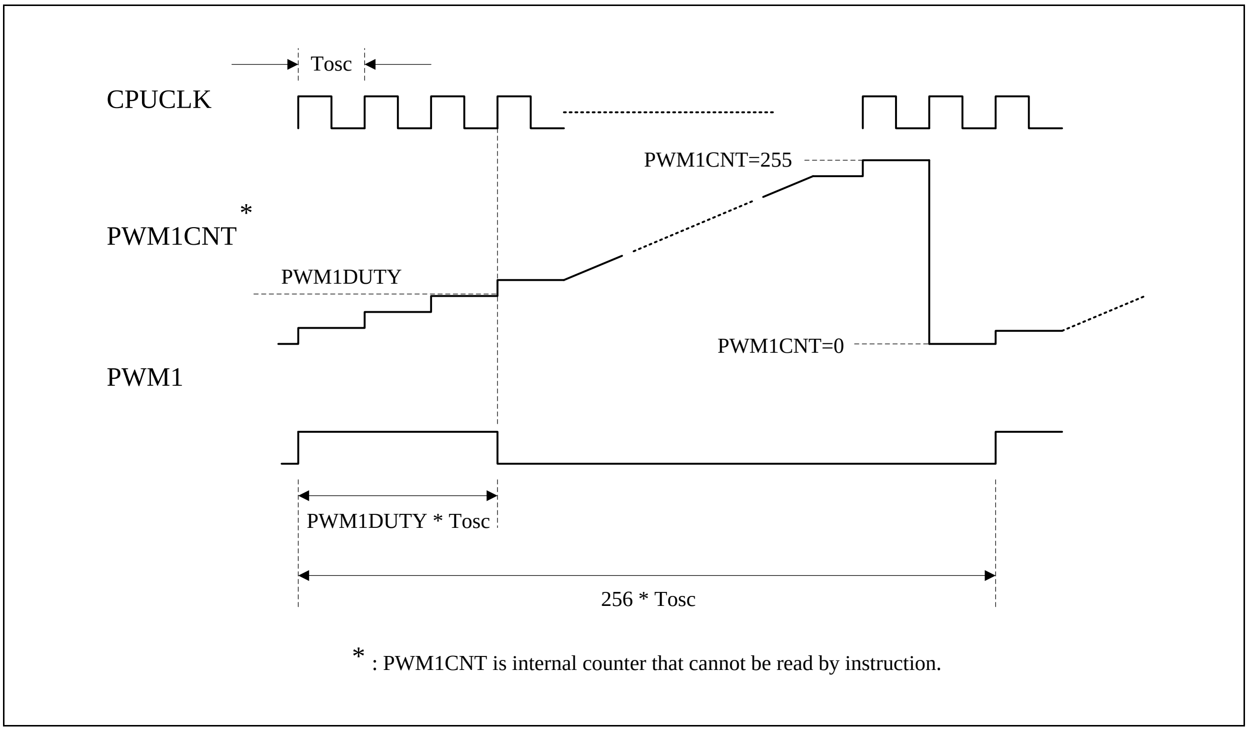


Figure 3.7.2 PWM1 Timing

3.8 UART

TM57FLA80 has built-in standard UART (Universal Asynchronous Receiver/Transmitter), which is responsible for performing serial communications among computers. Transmitting device changes the parallel data to serial data and sends the bit-stream data via TX line, the receiving device receives the data via RX line in serial form and converts to parallel data stored in register for CPU reading. The TX/RX module also handles data synchronizations, parity bit generation and detection, frame error, overrun error, and interrupt generation. Fig 3.8.1 shows the data format of UART.

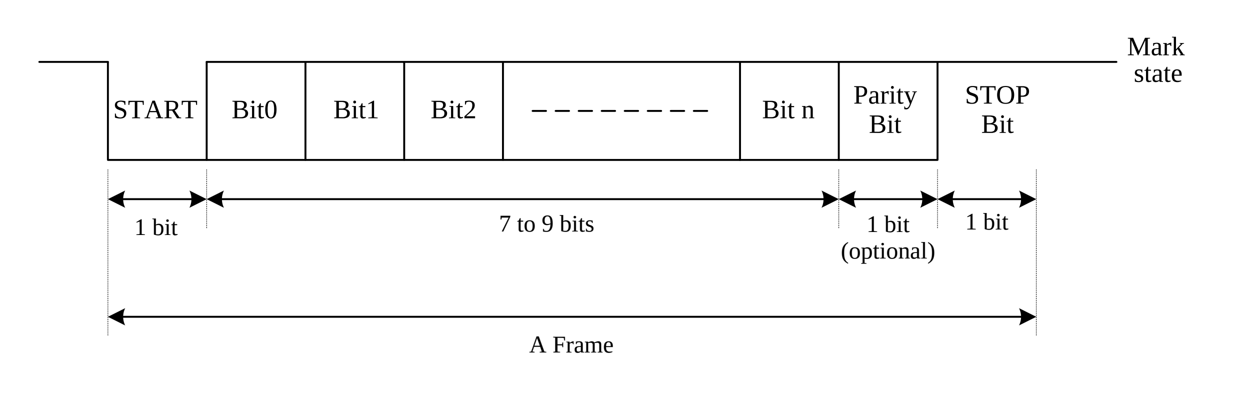


Figure 3.8.1 UART data format

The transmission line is normally kept in mark state (logic 1) until the transmission or reception start with a transition to space state (logic 0). The first bit transmitted is START bit and the length is 1 bit wide (1 bps second), followed by the transmitted bit stream of the data. The LSB (Least Significant Bit) is sent first, the number of transmitted bits is defined by UMODE control bits that can have 7, 8, or 9 bits mode to be transmitted or received. The parity bit is followed by the data bit and it is selectable to be sent or not, but in 9-bit mode transmission, the parity bit is always not to be transmitted.

UART Modes

The UART can operate in one of three modes, e.g. Mode 0 (7-bit data), Mode 1 (8-bit data), and Mode 2 (9-bit data). The data format of Mode 0 and Mode 1 can transmit/receive parity bit according to PRE bit. If PRE=1, the TX data format will add the parity bit after the data bits, otherwise, the TX only transmits the data bits. Mode 2 does not support parity bit transmit/receive. Note that the setting of UART between two communicating devices must be the same so that the RX part can check the data format and parity the same as TX part. The parity select bit is EVEN bit in UARTS control register. It uses even parity if EVEN=1, else it uses odd parity. Every UART transfer is ended with a STOP bit. If the STOP bit is not seen by RX part, the RX part will set the FMERR bit in UARTS control register to indicate the transfer may not be properly received/transmitted, it might be re-sent again or firmware handling. Figure 3.8.2 shows the data formats of three UART modes.

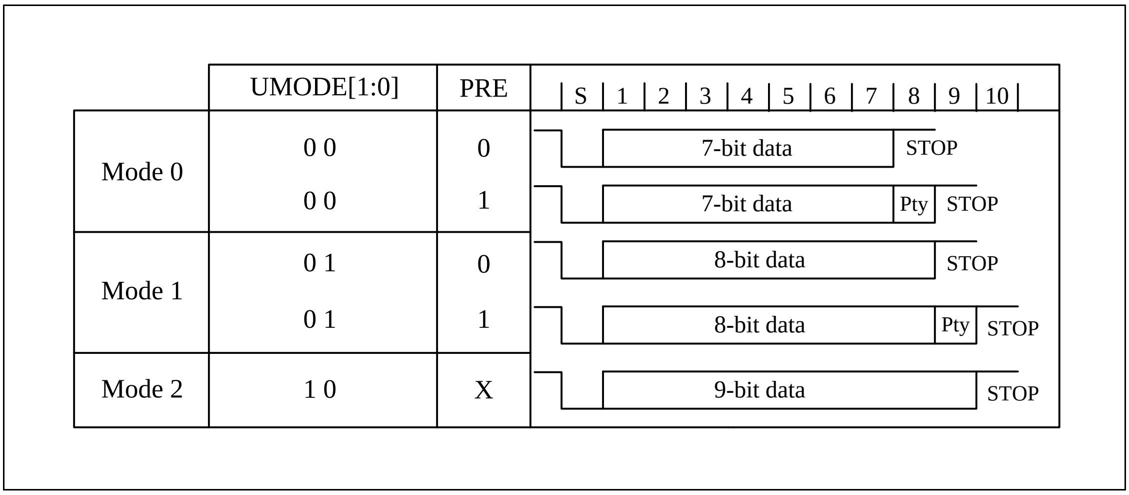


Figure 3.8.2 UART Modes

UART System Blocks

Figure 3.8.3 shows the system blocks of the UART built in TM57FLA80. The Baud Clock Generator divides the Fosc to produce the proper frequency of the UART baud rate. The divisor is UBAUD register in R-plane. Every bit of data stream is sampled by 16 Baud clocks which are generated from Baud Clock Generator. The desired Baud rate can be achieved by setting UBAUD with the following equation:

$$\text{Baud Rate} = \frac{F_{CPUCLK}}{2 * 16 * UBAUD}$$

The divided clock (Baud clock) is then sent to TX and RX control logic, and TX/RX shifter buffers. The TX control logic start sending data out to TX pin (PG4) once the URDATA is written, URTD9 will be sent if the UART mode is Mode 2. When transmit completes, the “TX Buffer Empty” signal is asserted, then interrupt will be generated if INTE2.URTIE is set. UINVEN will invert the output when it is set to 1, the RX will be inverted when UINVEN is set.

The UART RX signal is input from RX pin (PG5), and then it is inverted if UINVEN is set, passing through a synchronization circuit then clocked to the RX shifter. Number of bits to be shifted into RX shifter depends on what UART Mode is operated, the RX control logic will handle them, including frame error detection (FMERR bit), parity detection (PRERR), and overrun error (OVERR) detection. Frame error means that the incoming frame STOP bit is not detected. Parity error means that the incoming data parity bit does not equal to the parity of data bits that are evaluated by the RX control logic. Overrun error means the previous transfer that is stored in URDATA has not been read as a new incoming transfer is completed, the old URDATA will be lost when overrun error occurs.

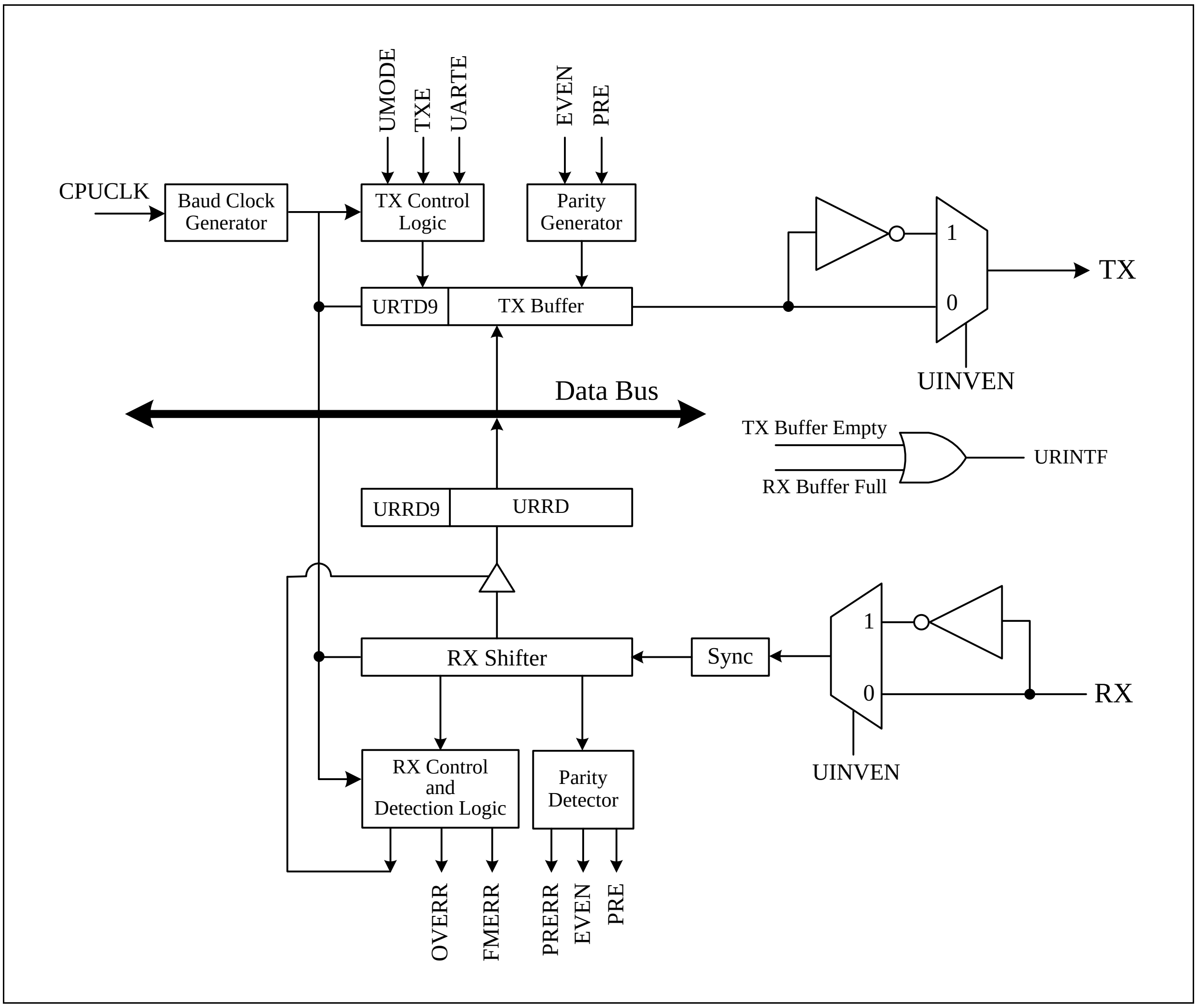


Figure 3.8.3 UART Block Diagram

Table 3.8.1 lists the common used operating frequency to generate common used Baud Rates. Some Baud rates cannot be achieved under some certain frequencies because the frequency error is too large to use.

Common Crystal (Hz)	Desired Baud Rate (bps)	UBAUD	Number of UBAUD bit required	Actual Generate Baud Rate	Frequency Error (%)	
4000000	1200	104	7	1201.9	0.16%	
6000000	1200	156	8	1201.9	0.16%	
8000000	1200	208	8	1201.9	0.16%	
10000000	1200	260	**9**	1201.9	0.16%	cannot use
12000000	1200	313	**9**	1198.1	0.16%	cannot use
4000000	2400	52	6	2403.8	0.16%	
6000000	2400	78	7	2403.8	0.16%	
8000000	2400	104	7	2403.8	0.16%	
10000000	2400	130	8	2403.8	0.16%	
12000000	2400	156	8	2403.8	0.16%	
4000000	4800	26	5	4807.7	0.16%	
6000000	4800	39	6	4807.7	0.16%	
8000000	4800	52	6	4807.7	0.16%	
10000000	4800	65	7	4807.7	0.16%	
12000000	4800	78	7	4807.7	0.16%	
4000000	9600	13	4	9615.4	0.16%	
6000000	9600	20	5	9375	2.34%	
8000000	9600	26	5	9615.4	0.16%	
10000000	9600	33	6	9469.7	1.36%	
12000000	9600	39	6	9615.4	0.16%	
4000000	19200	7	3	17857.1	6.99%	don't use
6000000	19200	10	4	18750	2.34%	
8000000	19200	13	4	19230.8	0.16%	
10000000	19200	16	4	19531.3	1.73%	
12000000	19200	20	5	18750	2.34%	
4000000	38400	3	2	41666.7	8.51%	don't use
6000000	38400	5	3	37500	2.34%	
8000000	38400	7	3	35714.3	6.99%	don't use
10000000	38400	8	3	39062.5	1.73%	
12000000	38400	10	4	37500	2.34%	
11059200	1200	288	**9**	1200	0.00%	cannot use
11059200	2400	144	8	2400	0.00%	
11059200	4800	72	7	4800	0.00%	
11059200	9600	36	6	9600	0.00%	
11059200	19200	18	5	19200	0.00%	
11059200	38400	9	4	38400	0.00%	
11059200	57600	6	3	57600	0.00%	
11059200	115200	3	2	115200	0.00%	

Table 3.8.1 UART Baud Rates setting under some common used frequencies

3.9 Serial Peripheral Interface (SPI)

The SPI module is capable of full-duplex, synchronous, serial communication between MCU and peripheral devices. The peripheral devices can be other MCUs, A/D converter, sensors, or flash memory, etc. The SPI runs at a baud rate up to the system clock divided by two. Firmware can read the status flags, or the operation can be interrupt driven.

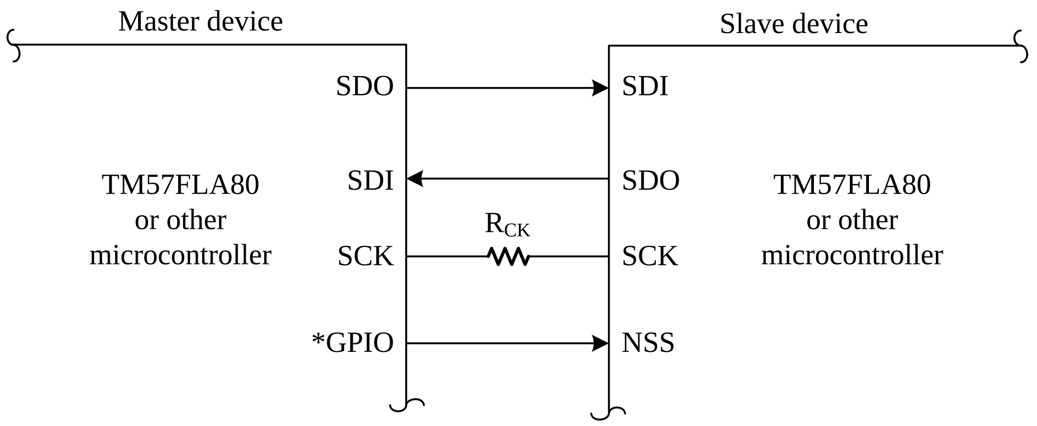
The features of the SPI module include:

- Master or Slave mode operation
- Full-duplex operation
- Programmable transmit bit rate
- Single-buffer receive
- Serial clock phase and polarity options
- MSB-first or LSB-first shifting selectable.

Figure 3.9.1 is the connection diagram between SPI master device and slave device. The master device initiates all SPI data transfers. During a transfer, the master shifts data out (SDO pin) to the slave while shifting data in (on SDI pin) from the slave at the same time. The transfer effectively exchanges the data that are in the SPI shift registers of the two SPI devices. SCK signal is a clock output from the master and input to the slave. The slave device selected by NSS pin goes low. NSS pin is necessary when it is slave mode. The rest 3 pins (SCK, SDI, SDO) are dedicated to SPI module when SPI function is enabled (SPE=1, SPEB=0).

Note that it is recommended that a resistor R_{CK} should be added between devices' SCK pins. By adding this resistor will improve the signal quality and reduce the noise on the lines especially when TM57FLA80 is used as a slave device. The typical value of R_{CK} is about 100 ohms.

It is important to note that when TM57FLA80 acts as a Slave device, the Master should ignore the first byte that exchanges from the Slave of each transfer. A transfer is defined that the NSS is low until the last byte is completely transferred.



* GPIO means General Purposed I/O pin

Figure 3.9.1 Connections between two SPI devices

SPI System Block

Figure 3.9.2 shows the SPI system block diagram. The oscillation clock Fosc is divided by SPIBR register. There are two 3-bit prescalers to generate desired SPI baud clock. The fastest bit rate is SPPR=000 (divided by 1), SPR=000 (divided by 2), it means one bit data duration is 2*(1/F_{CPUCLK}).

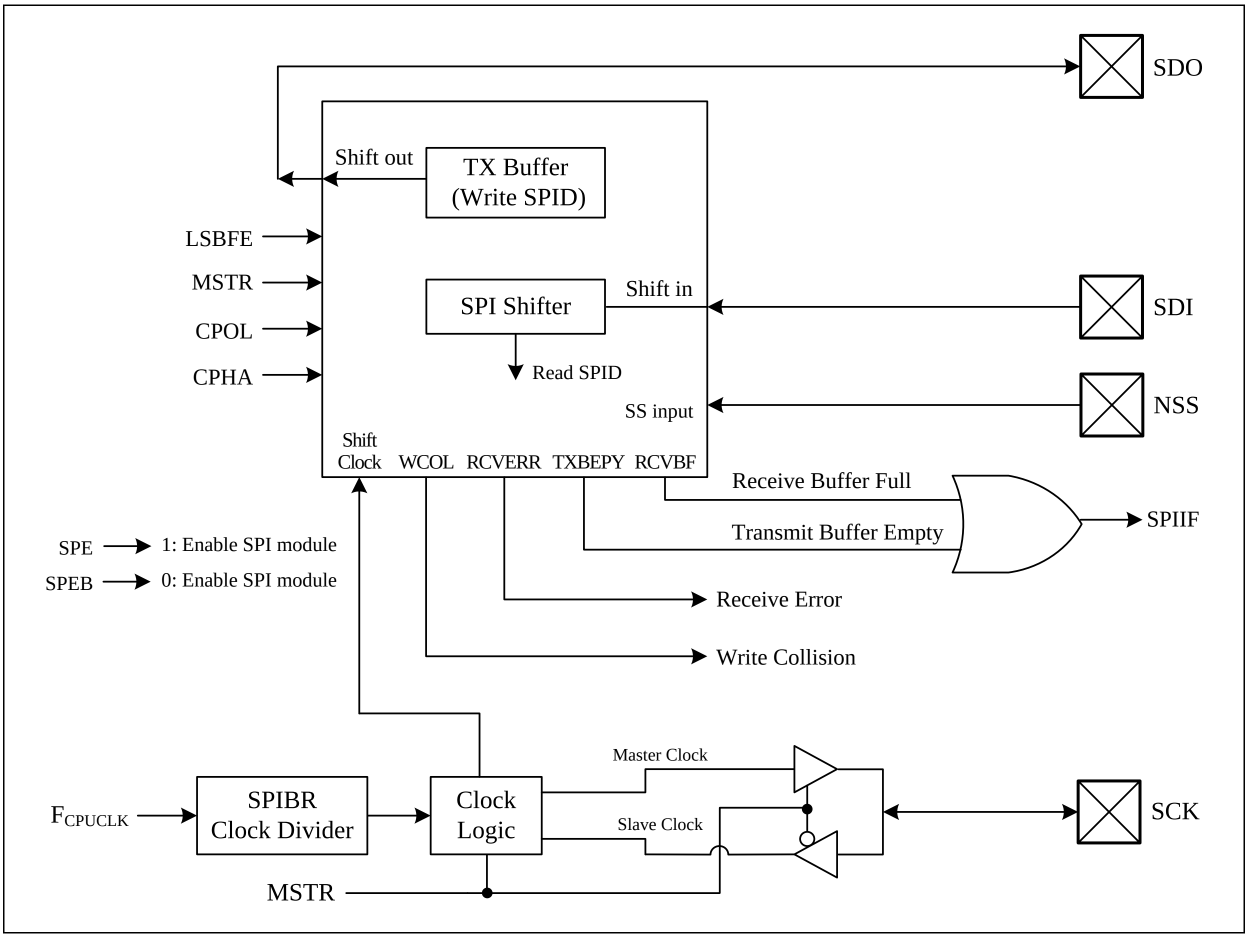


Figure 3.9.2 SPI System Block Diagram

The kernel module of the SPI is the SPI shifter. Data are written to the double-buffered transmitter (by writing to SPID) and then start the data transfer bit by bit out to SDO pin. At the mean time, the bit-stream data are received from SDI pin from the connected device. That is, after the transfer completes, the data of the master device and slave device are exchanged.

When the SPI is configured as a master, the clock output is SCK pin, the shifter output is SDO pin, the shifter input is SDI pin, and the slave select output is NSS pin. Note that the TX buffer is not actually shifted out, after the transfer completes, the content of internal TX buffer will keep its value until instruction writes SPID again. The received data byte can be read by instruction reads SPID register. The two internal registers are physically different even they share one name and F-plane address.

If the connected device is TM57FLA80 slave device, the master should ignore the first byte exchanged from the slave in each transfer. If the slave device is other chip such as SPI flash, user must refer to their datasheet for detail access methods.

When the SPI is configured as a slave, the clock input is SCK pin, the shifter output is SDO pin, the shifter input is SDI pin, and the slave select input is NSS pin. The mechanisms are the same as master device which is described above. With using as slave device, it is strongly recommended that a 100 ohms resistor is connected in series between SCKs to reduce the clock noise.

If the system contains one master and at least 2 slaves, SDO pin of master connects to SDI pins of all slaves, SDI pin of master connects to SDO pins of all slaves, SCK pins are connected each other. NSS pins of slave should be connected by I/O pins of the master. Master device should use I/O pins to access these slave devices by software.

Functional Description

An SPI transfer is initiated by checking for the SPI transmit buffer empty flag (TXBEPY=1) and then writing a byte of data to the SPI data register (SPID) in the master device. The SPI transfer is immediately started to send out the data. Although the transmit buffer is physically not the receive buffer, but, once the SPID is written a byte of data to start transmitting, the receive buffer will be updated by the transfer to exchange the data stored in the slave device.

During the SPI transfer, data are sampled (read) on the SDI pin at one SCK edge and shifted, changing the bit value on the SDO pin, one-half SCK cycle later. After eight SCK cycles, the data that are in the TX buffer have been shifted out the SDO pin to the slave while eight bits of data are shifted in the SDI pin into the master's RX shifter. After the end of this transfer, the received data byte is moved from the shifter into the read buffer (instruction reads SPID) and RCVBF is set to indicate the data can be read by reading SPID. If another byte of data is written to SPID at the end of a transfer, a new transfer is started. If the SPID is not read before the new transfer completes, the write collision occurs (WCOL=1). RCVBF will be cleared automatically when executing read SPID, even the incoming data byte will not be used by master device, it is suggested to read SPID after each transfer completes to prevent WCOL=1, WCOL flag can be cleared by software.

Normally, the SPI module transfers MSB first. If the LSBFE=1, SPI data are shifted LSB first.

When the SPI is configured as a slave, its NSS pin must be driven low before a transfer starts and NSS must keep low during the transfer.

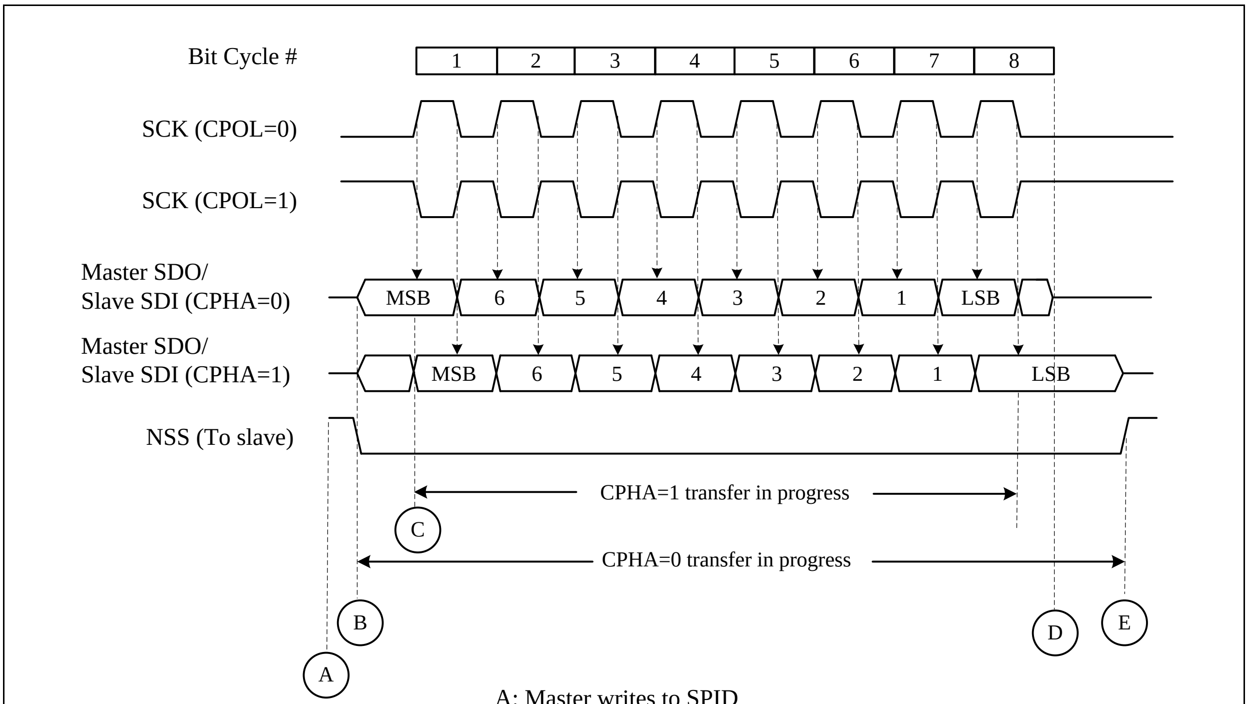
In the case of a write collision occurs (WCOL=1), the new data are lost because the RX shifter still holds the previous byte of data and is not ready to accept the new data. Software must read the data as soon as possible to prevent data collision.

SPI Clock Format

Figure 3.9.3 shows the four modes of SPI clock and data format. NSS will be asserted low while a transfer is started. CPOL is the SCK pin priority select. SCK pin in a logic high state while not transferring if CPOL=1, on the other hand, SCK pin in a logic low while not transferring if CPOL=0.

When CPHA=0, the first edge of SCK samples$_{\text{(the slave does)}}$ the MSB into RX shifter. The successive bit$_{\text{(bit6)}}$ is placed on master's SDO pin at the second edge of SCK. In summary, when CPHA=0, the slave samples the data bits on odd number(1, 3, 5, 7…) of SCK clock edge, and the even number(2, 4, 6, 8…) of SCK edge is the data preparation time of the master.

When CPHA=1, The SDO pin is in unknown level until the first edge of SCK coming. When the first edge of SCK coming, the master places the MSB on SDO. The slave uses the second edge of SCK to sample the MSB into RX shifter. The successive bit (bit6) is placed on master's SDO pin at the third edge of SCK. In summary, when CPHA=1, the slave samples the data bits on even number(2, 4, 6, 8…) of SCK clock edge, while the odd number(1, 3, 5, 7…) of SCK clock edge is the data preparation time of the master. Note that the LSB will remain on SDO pin until the NSS pin is deasserted to logic 1.



A: Master writes to SPID
B: NSS asserted automatically (if SSOE=1)
C: First SCK edge
D: Last bit of data, SPIIF is set
E: NSS deasserted

Figure 3.9.3 SPI Transfer Format

3.10 LCD

The chip has the LCD (Liquid Crystal Display) driver. It is capable of driving the LCD panel with max 160 dots with 8 Commons and 20 Segments. Figure 3.10.1 shows the block diagram of LCD. The VLCD is divided from V_{CC} by a set of resistors with the brightness selection bits to vary the VLCD from (3/5) V_{CC} to V_{CC}. There are totally 7 voltages, (1, 3/4, 2/3, 1/2, 1/3, 1/4, 0) * VLCD, respectively. The two MUXs use these 7 voltages to generate the individual COM and SEG driving waveform.

The LCDCLK is generated from CPUCLK or Slow Clock depends on TIMER2CLK bit because the LCDCLK comes from Timer2.

If the duty is set to static, only COM0 is the active COM line, while the even numbers (i.e. SEG0, SEG2, SEG4,…, SEG22) of SEG lines are active.

If the duty is set to 1/3 or 1/4 duty, the addresses of the LCDRAM are only used from 20h to 2Bh, and the SEG lines can be used up to 24. If the duty is set to 1/8 duty, the address of the LCDRAM ranges from 20h to 33h, and the SEG lines can be used up to 20. See the Table 3.10.1 for detail LCD RAM map.

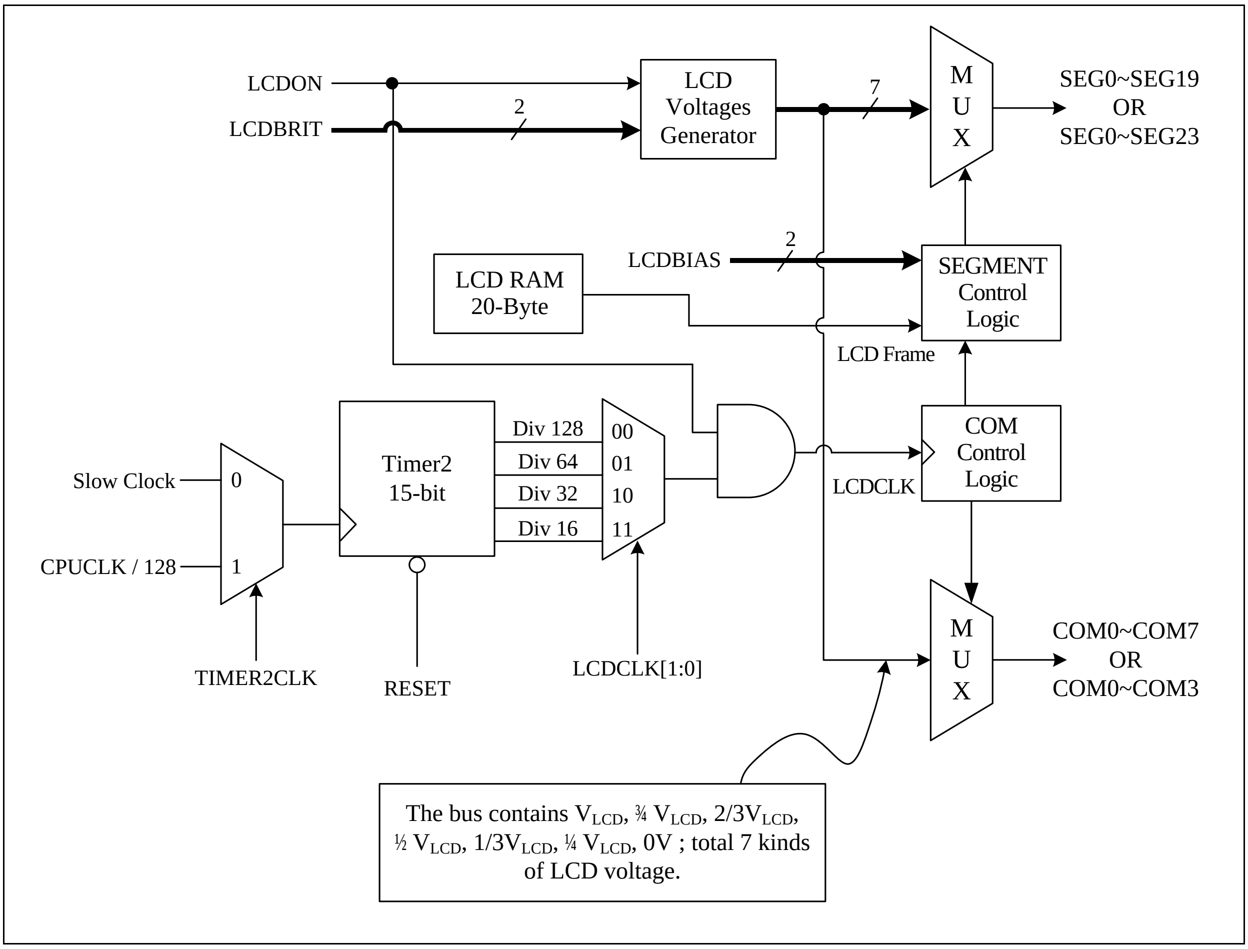


Figure 3.10.1 Block diagram of LCD driver.

8 COM	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0
R-Plane 20	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0
21	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1
22	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2
23	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3
24	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4
25	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5
26	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6
27	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7
28	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8
29	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9
2a	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10
2b	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11
2c	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12
2d	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13
2e	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14
2f	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15
30	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16
31	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17
32	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18
33	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19

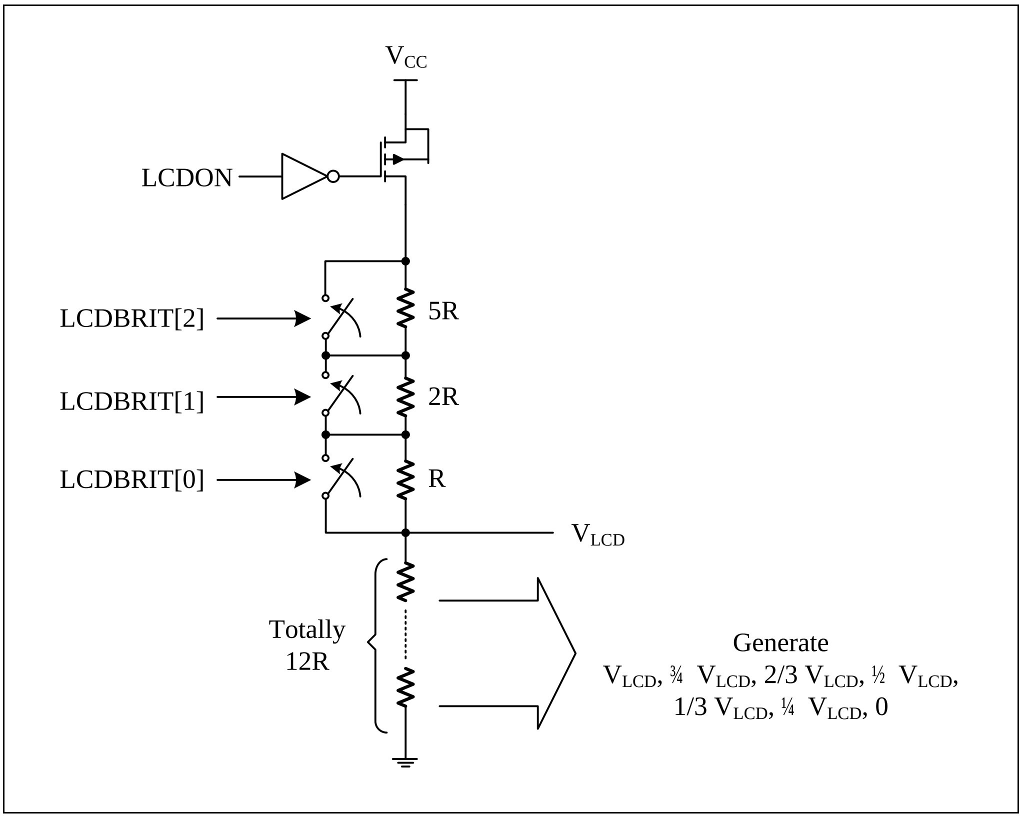
3,4 COM	COM3	COM2	COM1	COM0	COM3	COM2	COM1	COM0
20	SEG1	SEG1	SEG1	SEG1	SEG0	SEG0	SEG0	SEG0
21	SEG3	SEG3	SEG3	SEG3	SEG2	SEG2	SEG2	SEG2
22	SEG5	SEG5	SEG5	SEG5	SEG4	SEG4	SEG4	SEG4
23	SEG7	SEG7	SEG7	SEG7	SEG6	SEG6	SEG6	SEG6
24	SEG9	SEG9	SEG9	SEG9	SEG8	SEG8	SEG8	SEG8
25	SEG11	SEG11	SEG11	SEG11	SEG10	SEG10	SEG10	SEG10
26	SEG13	SEG13	SEG13	SEG13	SEG12	SEG12	SEG12	SEG12
27	SEG15	SEG15	SEG15	SEG15	SEG14	SEG14	SEG14	SEG14
28	SEG17	SEG17	SEG17	SEG17	SEG16	SEG16	SEG16	SEG16
29	SEG19	SEG19	SEG19	SEG19	SEG18	SEG18	SEG18	SEG18
2a	SEG21	SEG21	SEG21	SEG21	SEG20	SEG20	SEG20	SEG20
2b	SEG23	SEG23	SEG23	SEG23	SEG22	SEG22	SEG22	SEG22

Table 3.10.1 LCD RAM map

Static Driving

V_{LCD} and the Voltage Divider, adjusting the brightness

Figure 3.10.2 shows the internal voltage divider composed by resistors. LCDON controls the current flows from V_{CC} to ground. If LCDON=0, the PMOS will turn off the path so that all LCD voltages will be 0V. If LCDON=1, the resistor divider will work to generate the 7 voltages to provide the LCD control module for generating the desired waveforms. LCDBRIT control bits will open/short the switches to determine the V_{LCD}, Table 3.10.2 shows the LCDBRIT versus corresponding V_{LCD}. The voltage divider circuit will consume current from 10 μA to 25 μA because the DC path is always on when LCDON=1.



V_{CC}
LCDON
LCDBRIT[2]
LCDBRIT[1]
LCDBRIT[0]
5R
2R
R
V_{LCD}
Totally 12R
Generate
V_{LCD}, ¾ V_{LCD}, 2/3 V_{LCD}, ½ V_{LCD},
1/3 V_{LCD}, ¼ V_{LCD}, 0

Figure 3.10.2 The LCD Voltage Divider

The higher V_{LCD} is, the higher V_{RMS} (Root Mean Square Voltage) is applied on the LCD segment, and it means the higher brightness. Use the proper setting of LCDBRIT to fit the LCD panel specification.

LCDBRIT	V_{LCD}
000	(12/20) * V_{CC}
001	(12/19) * V_{CC}
010	(12/18) * V_{CC}
011	(12/17) * V_{CC}
100	(12/15) * V_{CC}
101	(12/14) * V_{CC}
110	(12/13) * V_{CC}
111	V_{CC}

Table 3.10.2 The V_{LCD} corresponding to LCDBRIT

The Waveforms of COM and SEG for different BIAS/DUTY

Static Drive:

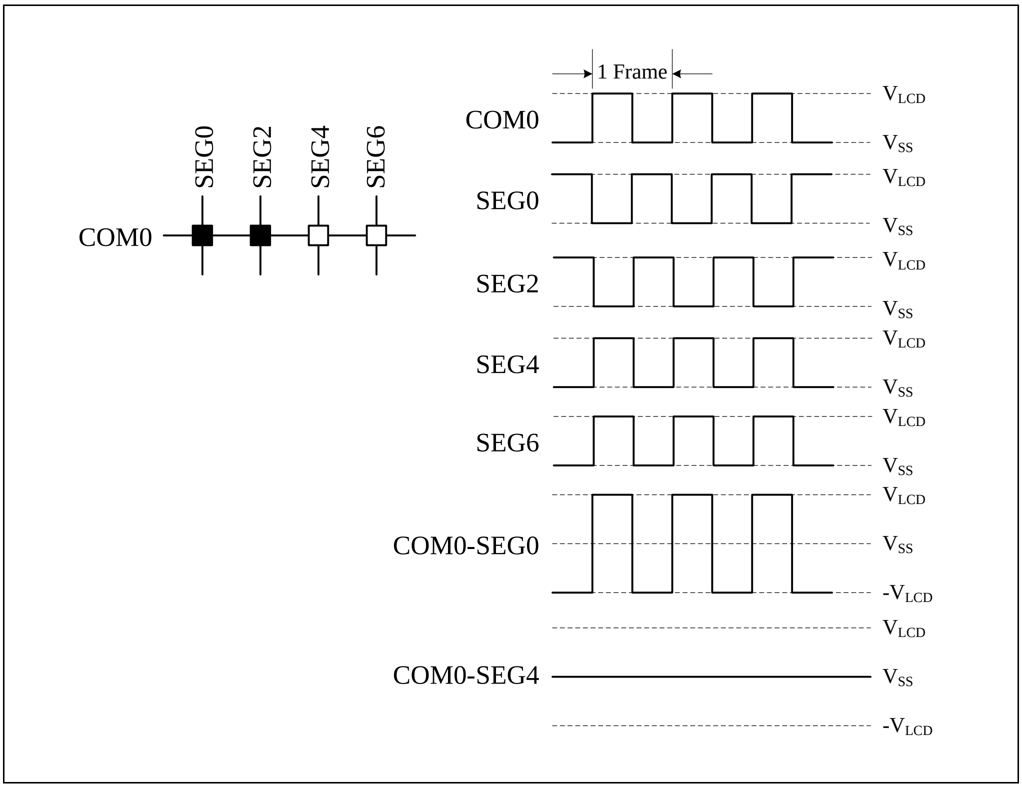


Figure 3.10.3 Static drive

1/4 Duty, 1/2 Bias Drive:

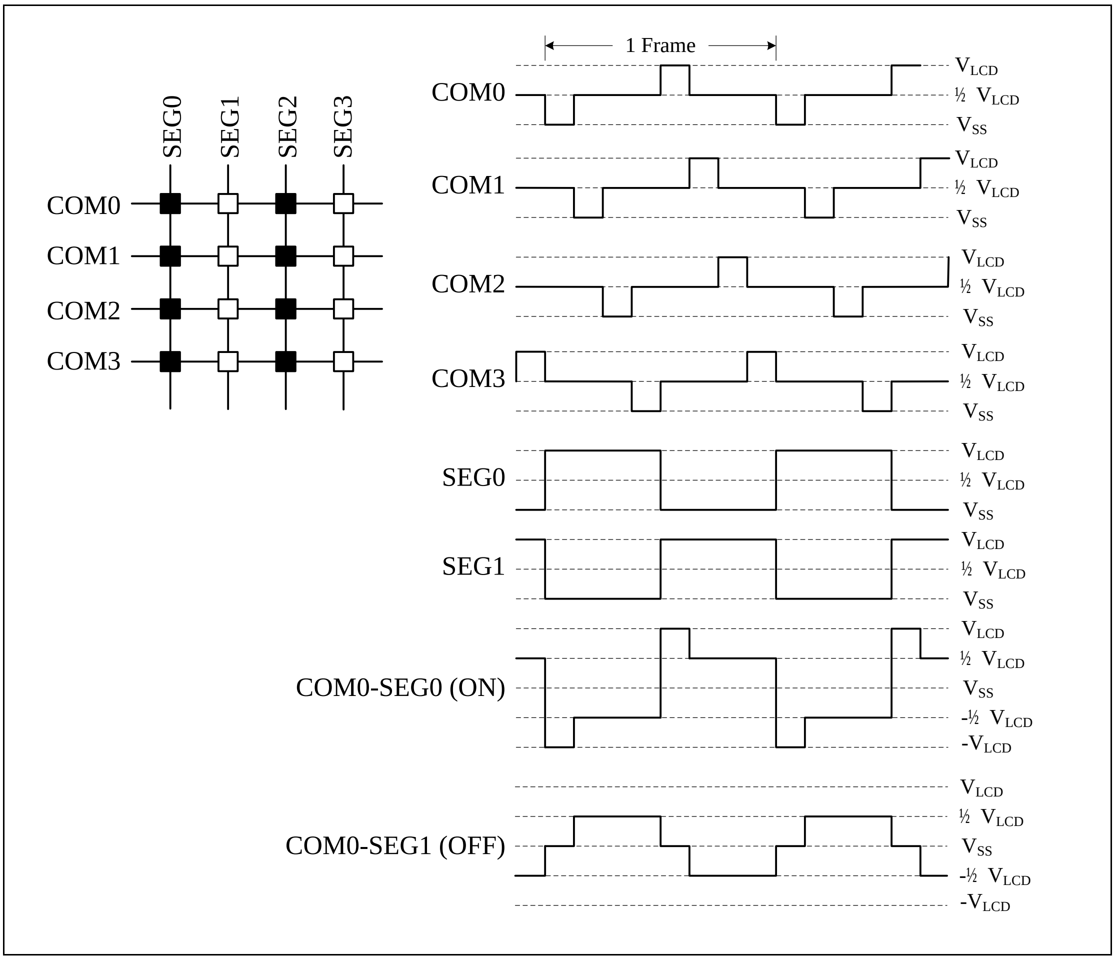


Figure 3.10.4 Configured as 1/4 Duty, 1/2 Bias

1/4 Duty, 1/3 Bias Drive:

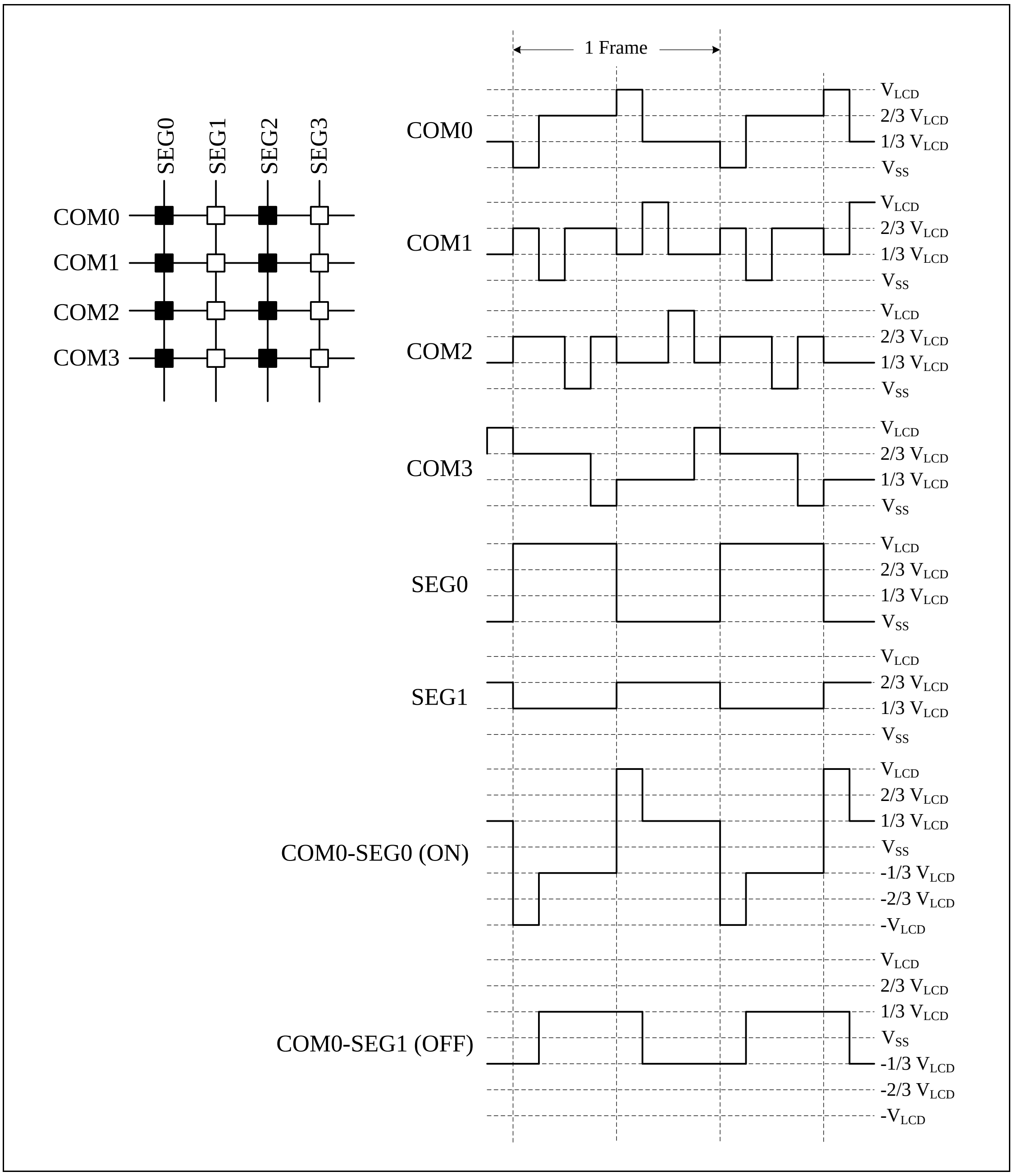


Figure 3.10.5 Configured as 1/4 Duty, 1/3 Bias

1/8 Duty, 1/4 Bias:

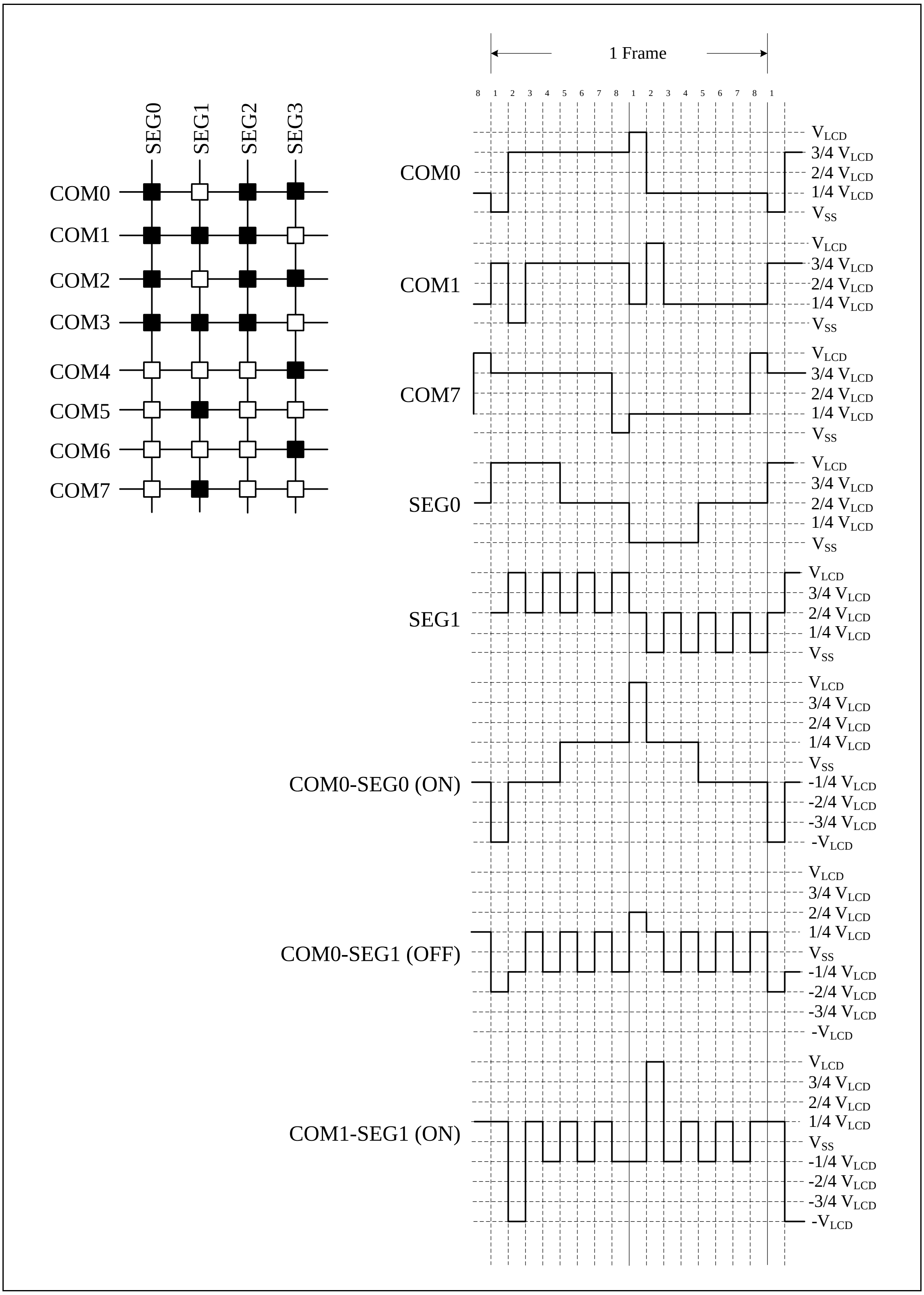


Figure 3.10.6 Configured as 1/8 Duty, 1/4 Bias

3.11 A/D Converter

Figure 3.11.1 shows the 12-bit ADC (Analog to Digital Converter) block diagram. The ADC consists of a 6-channel analog input multiplexer, control registers, clock generator, 12-bit successive approximation register, and output data registers. To use the ADC, user needs to set the ADCLKS to choose a proper ADC clock frequency, which must be less than 2 MHz. User then launches the ADC conversion by setting the ADCSTART control bit. After end of conversion, H/W clears the ADCSTART bit automatically. User can acquire the conversion status by polling this bit. The ADCPIN control register is used for ADC pin type setting, write the corresponding bit to "0" when the pin is used as an ADC input. It can disable the internal input Schmitt trigger by setting the pin for ADC input to save power consumption.

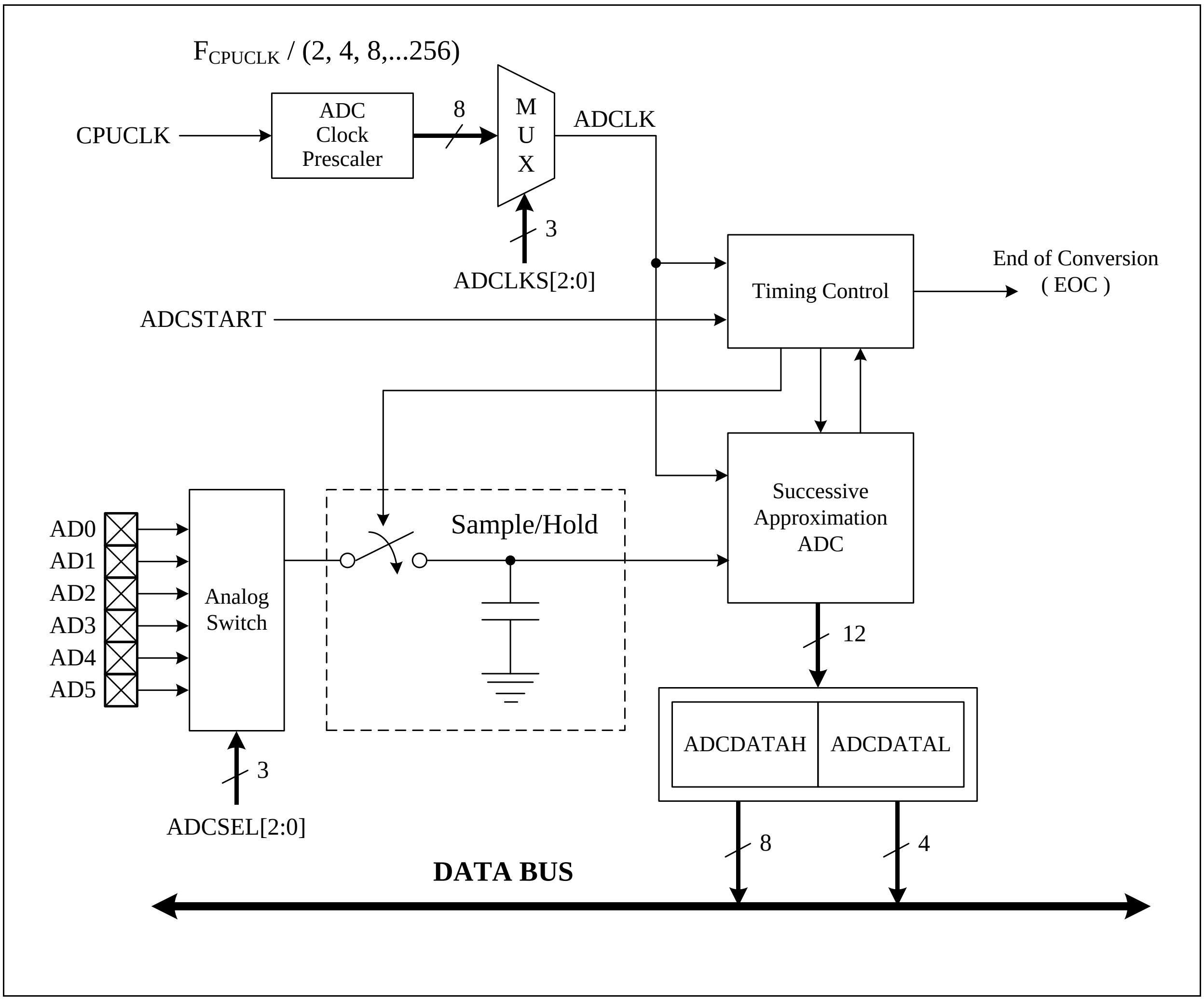


Figure 3.11.1 ADC Block Diagram

Figure 3.11.2 shows the internal ADC conversion timing. The conversion is completed after 50 ADC clocks since ADCSTART is asserted. ADCSTART will be cleared to “0” at the end of conversion.

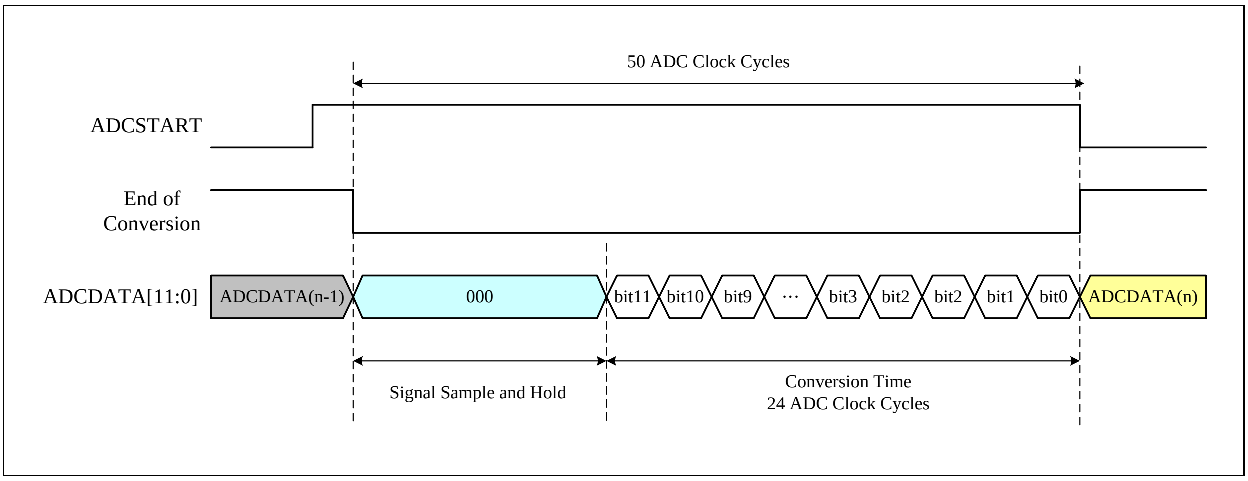


Figure 3.11.2 The ADC conversion timing

ADC example code

```
          Movlw     00000101b
          movwf     11h            ;ADC channel select,AD5 (PA0)(ADCSEL)
          movlw     00000001b
          movwr     08h            ;disable PA0 pull up resistor (PAPU)
          movlw     11011111b
          movwr     16h            ;set AD5 input enable (ADPIN)
          movlw     00100000b
          movwr     0ch            ;set ADC clock is Fosc/64   (ADCLKS)
          bsf       11h, 3         ;start ADC conversion (ADCSTART)

     ADC_LOOP:
          btfsc     11h, 3
          goto      ADC_LOOP       ;wait ADCSTART go low
          :                        ; read ADCQ[11:0]   (ADCH, ADCL)
          :
```

3.12 Touch Key

As noted in section 3.2 (Timer0), the Touch Key Module outputs the oscillation clock to Timer0 and counts like T0I input. Figure 3.12.1 shows the block diagram of the Touch Key module. It consists of an RC oscillator, 16-to-1 analog input select, TKSPEED control bits select the output of the frequency divider. The frequency divider divides the oscillation clock by 2, 8, 32, and 128. If TKE bit is 1, the divided clock will be sent to Timer0 to count at the rising or falling edge depends on T0IEDGE bit.

If the human finger tips close to the touch pad, the equivalent capacitance of C will be increased, that is, the oscillation frequency will be decreased.

Based on the above thesis, user program needs to observe what input channel causes the lowest Timer0 counting value in a fixed period of time, which channel of key is touched or the finger is just approaching.

To distinguish what channel counting value is the lowest, we need another Timer (Timer1, Timer2, or WKT Timer) to set up a proper interval of time that Timer0 will not count to overflow. Based on this fixed time interval, the user program switches the Touch Key channels one after another and finds the lowest value of Timer0, which is the key in touching or approaching.

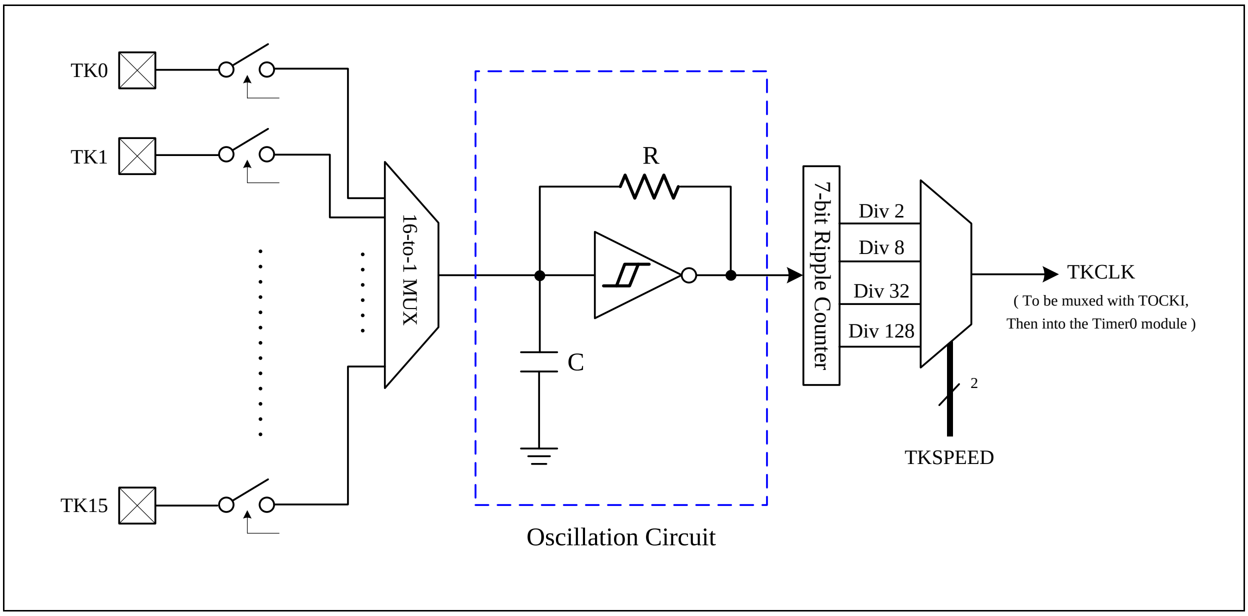


Figure 3.12.1 Touch Key Oscillator Block Diagram

Figure 3.12.2 shows the flowchart how to use Timer0 and Timer1 to determine which channel of the Touch Key is pressed. Using the 16-bit Timer1 to set up a fix interval of time and utilize the Timer1 interrupt to stop Timer0 and store its value if it is not overflow. Determine the lowest value of Timer0 of the desired channels, which is the key pressed.

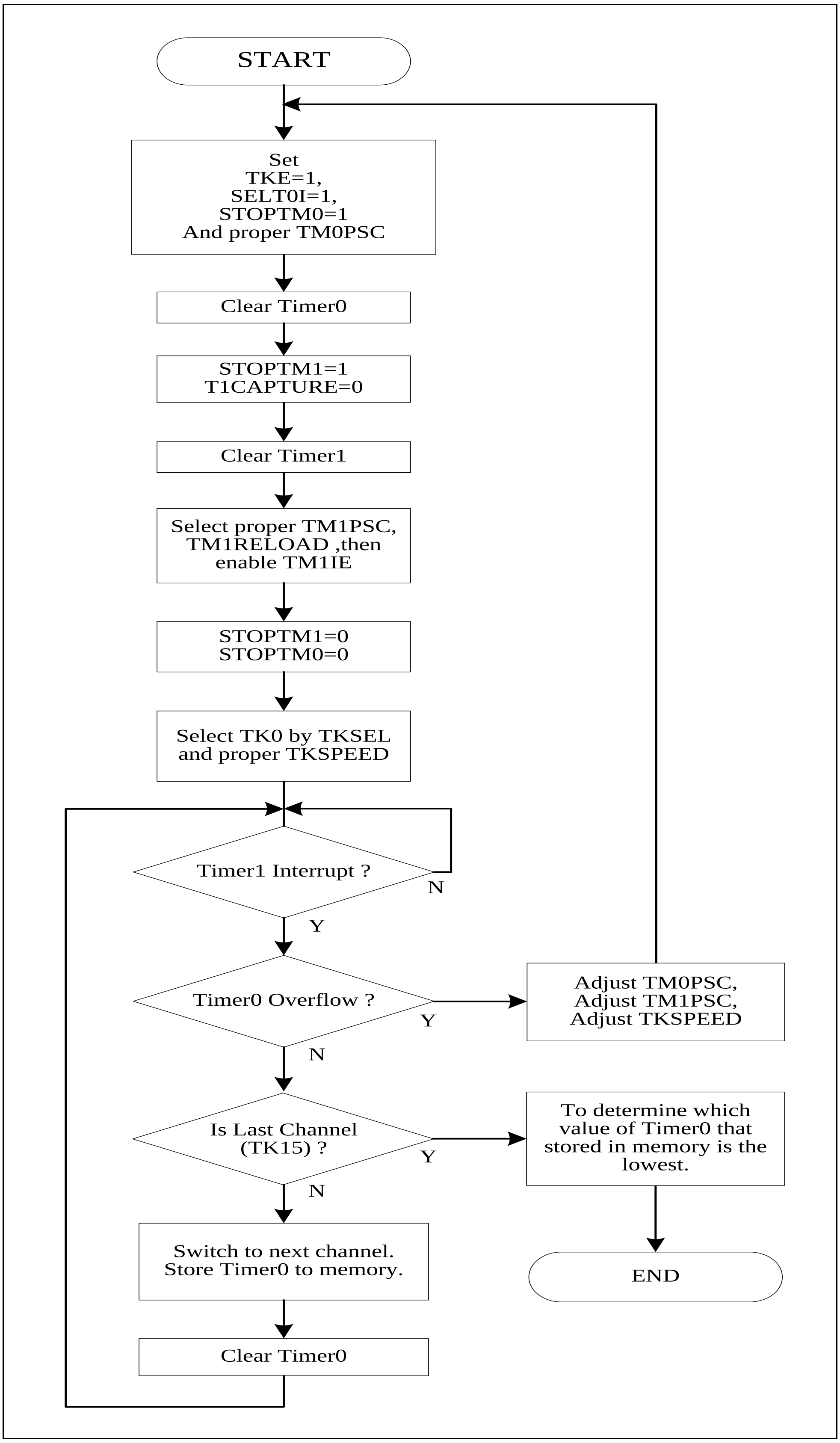


Figure 3.12.2 The recommended procedures for using the Touch Key function

3.13 System Clock Oscillator

System Clock can be operated in five different oscillation modes, which can be selected by setting the CLKS in the SYSCFG register and SELSUB, SUBXRC, SUBCKE, and STOPFCK control bits in CLKCTRL register. In Fast/Slow Crystal mode, a crystal or ceramic resonator is connected to the FXI (or SXI) and FXO (or SXO) pins to establish oscillation. In Fast/Slow External RC mode, the external resistor and capacitor determine the oscillation frequency. In the FIRC mode, the on-chip oscillator generates 4 MHz system clock. In this mode, PCB Layout may have strong effect on the stability of Internal Clock Oscillator. Since power noise degrades the performance of Internal Clock Oscillator, placing power supply bypass capacitors 1 uF and 0.1 uF very close to Vcc /VSS pins improves the stability of clock and the overall system. Figure 3.13.1 shows the external components need to be connected with Fast/Slow crystal modes and Fast/Slow External RC modes.

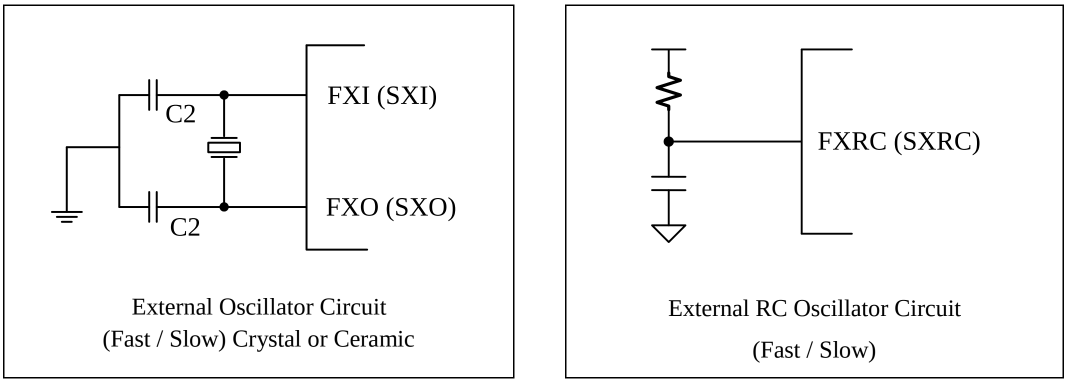
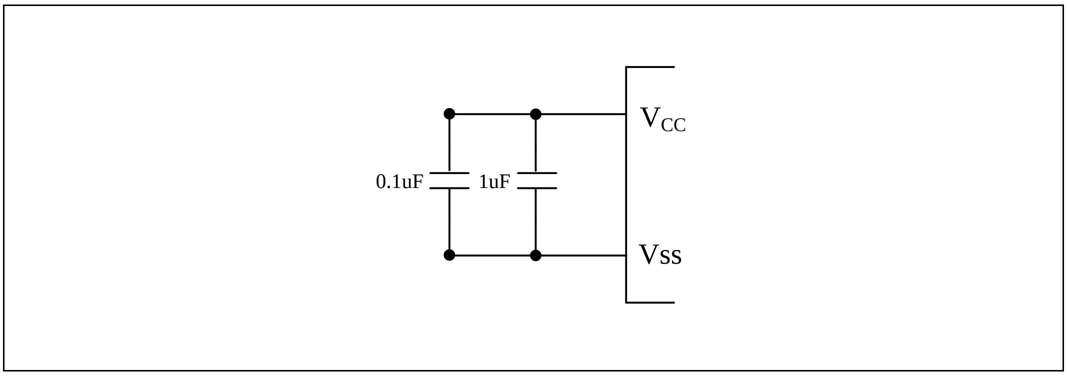

C2
C2
FXI (SXI)
FXO (SXO)
External Oscillator Circuit
(Fast / Slow) Crystal or Ceramic
FXRC (SXRC)
External RC Oscillator Circuit
(Fast / Slow)

Figure 3.13.1 Oscillation Circuits



Internal RC Mode

MEMORY MAP

F-Plane

Name	Address	R/W	Rst	Description
INDF	00.7~0	R/W	-	Not a physical register, addressing INDF actually point to the register whose address contains in the FSR register.
TIMER0	01.7~0	R/W	0	Timer 0
PC	02.7~0	R/W	0	Program Counter [7~0]
STATUS				
ROMPAGE	03.7	R/W	0	Program ROM Page Select
GBIT	03.6	R/W	0	General purpose bit
RAMBANK	03.5	R/W	0	RAM Bank Select
TO	03.4	R	0	WDT time out flag
PD	03.3	R	0	Sleep mode flag
ZFLAG	03.2	R/W	0	Zero Flag
DCFLAG	03.1	R/W	0	Decimal Carry Flag
CFLAG	03.0	R/W	0	Carry Flag
FSR	04.7~0	R/W	0	F-Plane File Select Register
PAD	05.6~0	R	-	Port A pin or "data register" state
		W	7f	Port A data output register
PBD	06.7~0	R	-	Port B pin or "data register" state
		W	ff	Port B data output register
RSR	07.7~0	R/W		R-Plane File Select Register
INTE1				Interrupt Enable Group 1, 1=Enable, 0=Disable
PWM0IE	08.7	R/W	0	PWM0 Interrupt Enable, 1=Enable, 0=Disable
TM2IE	08.6	R/W	0	Timer2 Interrupt Enable, 1=Enable, 0=Disable
TM1IE	08.5	R/W	0	Timer1 Interrupt Enable, 1=Enable, 0=Disable
TM0IE	08.4	R/W	0	Timer0 Interrupt Enable, 1=Enable, 0=Disable
WKTIE	08.3	R/W	0	Wakeup Timer Interrupt Enable, 1=Enable 0=Disable
XINT2E	08.2	R/W	0	XINT2 (PB1) falling Interrupt Enable, 1=Enable, 0=Disable
XINT1E	08.1	R/W	0	XINT1 (PB0) falling Interrupt Enable, 1=Enable 0=Disable
XINT0E	08.0	R/W	0	XINT0 (PA0) falling/rising Interrupt Enable, 1=Enable, 0=Disable

Name	Address	R/W	Rst	Description
INTF1				Interrupt Flag Group 1
PWM0I	09.7	R	-	PWM0 interrupt event pending flag, set by H/W while PWM0 overflows.
		W	0	write 0: clear this flag; write 1: no action.
TM2I	09.6	R	-	Timer2 interrupt event pending flag, set by H/W while Timer2 matches.
		W	0	write 0: clear this flag; write 1: no action.
TM1I	09.5	R	-	Timer1 interrupt event pending flag, set by H/W while Timer1 overflows.
		W	0	write 0: clear this flag; write 1: no action.
TM0I	09.4	R	-	Timer0 interrupt event pending flag, set by H/W while Timer0 overflows.
		W	0	write 0: clear this flag; write 1: no action.
WKTI	09.3	R	-	WKT interrupt event pending flag, set by H/W while WKT is timeout.
		W	0	write 0: clear this flag; write 1: no action.
XINT2	09.2	R	-	XINT2 pin falling interrupt pending flag, set by H/W at INT2 pin's falling edge. Belongs to XINTA interrupt.
		W	0	write 0: clear this flag; write 1: no action.
XINT1	09.1	R	-	XINT1 pin falling interrupt pending flag, set by H/W at INT1 pin's falling edge. Belongs to XINTA interrupt.
		W	0	write 0: clear this flag; write 1: no action
XINT0	09.0	R	-	XINT0 pin falling/rising interrupt pending flag, set by H/W at INT0 pin's falling/rising edge. Belongs to XINTA interrupt.
		W	0	write 0: clear this flag; write 1: no action
TM1L				
TIMER1L	0a.7~0	R		Timer 1 Counter low byte
TIMER1L	0a.7~0	W	0	Timer 1 reload data low byte
TM1H				
TIMER1H	0b.7~0	R		Timer 1 Counter high byte
TIMER1H	0b.7~0	W	0	Timer 1 reload data high byte
PWM0				
PWM0DUTY	0c.7~0	R/W	0	PWM0 duty 8-bit
TM1CTRL				
CLRTM1	0d.3	R/W	0	Timer1 clear and hold when this bit is "1".
CLRPWM0	0d.2	R/W	1	PWM0 clear and hold when this bit is "1".
STOPTM1	0d.1	R/W	0	Stop Timer1 when this bit is "1".
STOPTM0	0d.0	R/W	0	Stop Timer0 when this bit is "1".
INTE2				Interrupt Enable Group 2, 1=Enable, 0=Disable.
URTIE	0e.6	R/W	0	UART TX/RX Complete Interrupt Enable, 1=Enable, 0=Disable.
SPIIE	0e.5	R/W	0	SPI TX/RX Complete Interrupt Enable, 1=Enable, 0=Disable.
XINT7E	0e.4	R/W	0	XINT7 (PA1) falling Interrupt Enable, 1=Enable, 0=Disable.
XINT6E	0e.3	R/W	0	XINT6 (PB7) falling/rising Interrupt Enable, 1=Enable, 0=Disable.
XINT5E	0e.2	R/W	0	XINT5 (PB6) falling Interrupt Enable, 1=Enable, 0=Disable.
XINT4E	0e.1	R/W	0	XINT4 (PB3) falling Interrupt Enable, 1=Enable, 0=Disable.
XINT3E	0e.0	R/W	0	XINT3 (PB2) falling Interrupt Enable.

Name	Address	R/W	Rst	Description
INTF2				Interrupt Flag Group 2
URTIF	0f.6	R	-	UART interrupt event pending flag, set by H/W while TX/RX transfers complete.
		W	0	write 0: clear this flag; write 1: no action.
SPIIF	0f.5	R	-	SPI interrupt event pending flag, set by H/W while data exchange complete.
		W	0	write 0: clear this flag; write 1: no action.
XINT7	0f.4	R	-	XINT7 pin falling interrupt pending flag, set by H/W at INT7 pin's falling edge. Belongs to XINTB interrupt.
		W	0	write 0: clear this flag; write 1: no action.
XINT6	0f.3	R	-	XINT6 pin falling/rising interrupt pending flag, set by H/W at INT6 pin's falling/rising edge. Belongs to XINTB interrupt.
		W	0	write 0: clear this flag; write 1: no action.
XINT5	0f.2	R	-	XINT5 pin falling interrupt pending flag, set by H/W at INT5 pin's falling edge. Belongs to XINTB interrupt.
		W	0	write 0: clear this flag; write 1: no action.
XINT4	0f.1	R	-	XINT4 pin falling interrupt pending flag, set by H/W at INT4 pin's falling edge. Belongs to XINTB interrupt.
		W	0	write 0: clear this flag; write 1: no action.
XINT3	0f.0	R	-	XINT3 pin falling interrupt pending flag, set by H/W at INT3 pin's falling edge. Belongs to XINTB interrupt.
		W	0	write 0: clear this flag; write 1: no action.
ADCH				
ADCDATA	10.7~0	R		ADC output data MSB (Most Significant Byte)
ADCL, ADC				
ADCDATA	11.7~4	R		ADC output data LSB (Least Significant Bits)
ADCSTART	11.3	R/W	0	ADC Start, H/W clear after end of conversion
ADCSEL	11.2~0	R/W	111	ADC channel select 000: AD0 001: AD1 010: AD2 011: AD3 100: AD4 101: AD5
PDD	12.7~0	R	-	Port D pin or "data register" state
		W	ff	Port D data output register
PED	**13.7~0**	R	-	Port E pin or "data register" state
		W	ff	Port E data output register
PFD	14.7~0	R	-	Port F pin or "data register" state
		W	ff	Port F data output register
PGD	15.5~0	R	-	Port G pin or "data register" state
		W	ff	Port G data output register
PWM1DUTY	16.7~0	R/W	0	PWM1 duty 8-bit

Name	Address	R/W	Rst	Description
UARTC				UART Control Register
URTD9	17.7	R/W	0	Transmitted 9th bit
UMODE	17.6~5	R/W	0	UART Mode Selection, 00: 7-bit, 01: 8-bit, 10: 9-bit, 11: Reserved.
-	17.4	-	0	Reserved
TXE	17.3	R/W	0	1: Enable 0: Disable Transmission
RXE	17.2	R/W	0	1: Enable 0: Disable packet reception
UARTE	17.1	R/W	0	1:UART func enable 0:UART func disable
UINVEN	17.0	R/W	0	1:Eanble TX/Rx output/input Inversion 0:Disable Inversion
UARTS				UART Status Register. EVEN and PRE are control bits
UTBE	18.7	R	1	1: Tx buffer empty 0: Tx is transmitting
URRD9	18.6	R	0	Received 9th bit
EVEN	18.5	R/W	0	1: EVEN, 0:ODD Parity (both TX and RX)
PRE	18.4	R/W	0	1: Enable, 0:Disable Parity Addition (both TX and RX)
PRERR	18.3	R/W	0	Set to 1 when parity error occurs , clear to 0 by F/W.
OVERR	18.2	R/W	0	Set to 1 when overrun error occurs, clear to 0 by F/W.
FMERR	18.1	R/W	0	Set to 1 when frame error occurs, clear to 0 by F/W.
URBF	18.0	R	0	Set to 1 when 1 byte (or 9 bits) is received. Read URRD will clear to 0 automatically and then to receive the succeeding data without overrun error.
URDATA	19.7~0	R/W	0	UART Transmission/Reception Data Buffer (write to transmission shifter and read from reception buffer)
SPID	1a.7~0	R/W	0	SPI Transmission/Reception Data Buffer (write to transmission shifter and read from reception buffer)
SPIS				SPI Status Register
RCVERR	1b.7	R/W	0	Set to 1 when SPI SLAVE receives error, clear to 0 by F/W (SLAVE only).
RCVBF	1b.6	R	0	H/W set to 1 when SPI received buffer is full. F/W read SPID will clear to 0 automatically.
TXBEPY	1b.5	R	1	Set to 1 when SPI transmitted buffer is empty. Writing SPID will clear to 0 and start transmitting (master mode) or wait for master's clock and data (slave mode).
WCOL	1b.4	R/W	0	Slave SPIR write collision: if RCVBF=1 and Master issues a data exchange; 0: No write collision occurs
-	1b.3~0	R	0	Reserved. Read as 0000.
-	1c~1f	-	0	Unused Area (for future extension)
SRAM	20~7f	R/W	-	2 banks of internal SRAM

R-Plane

Name	Address	R/W	Rst	Description
INDR	00	R/W	-	Not a physical register, addressing INDR actually point to the register whose address contains in the RSR register.
OPTION				
CAPOLARITY	02.7	W	0	Timer0 Capture polarity. 0: High level capture, 1: Low level capture
T0CAPTURE	02.6	W	0	1: Timer0 works in CAPTURE Mode; 0: Timer0 works in COUNTER Mode
T0IEDGE/CAP	02.5	W	0	1: T0CKI/TKRC falling edge; 0: T0CKI/TKRC rising edge for Timer0 Prescaler count
SELT0I	02.4	W	0	1: T0CKI/TKRC as Timer0 Prescaler clock; 0: Instruction Cycle as Timer0 Prescaler clock
TM0PSC	02.3~0	W	0	Timer0 Pre-Scale 0000: div1 0001: div2 0010: div4 0011: div8 0100: div16 0101: div32 0110: div64 0111: div128 1xxx: div256
PWRDOWN	03	W		Write this register to enter Power-Down Mode.
CLRWDT	04	W		Write this register to clear WDT.
PAE	05.6~3	W	0	Each bit controls its corresponding pin, if the bit is 0: the pin is open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output
	05.2~0	W	0	Each bit controls its corresponding pin, if the bit is 0: the pin is pseudo-open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output
PBE	06.7~0	W	0	Each bit controls its corresponding pin, if the bit is 0: the pin is open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output
PWM0PRD	07.7~0	W	ff	PWM0 Period. ff=256
PAPUn	08.6~0	W	1	PA pull-up. 0: Enable 1: Disable
PBPUn	09.7~0	W	1	PB pull-up. 0: Enable 1: Disable
LCDPIN, PUN				
LCDPIN	0a.6~4	W	000	000: 0 pin, all I/O 001: 8 pin, COM[0~3], SEG[20~23] LCD; PE, PF, PG I/O 010: 12 pin, COM[0~7], SEG[16~19] LCD; PE[3:0], PF, PG I/O 011: 16 pin, COM[0~7], SEG[12~19] LCD, PF, PG I/O 100: 20 pin, COM[0~7], SEG[8~19] LCD, PF[3:0], PG I/O 101: 24 pin, COM[0~7], SEG[4~19] LCD, PG I/O 110: 26 pin, COM[0~7], SEG[2~19] LCD, PG[3:2] I/O 111: 28 pin, COM[0~7], SEG[0~19] LCD ; all LCD
PGPUn	0a.3	W	1	PG pull-up, 0=Enable 1:Disable
PFPUn	0a.2	W	1	PF pull-up, 0=Enable 1:Disable
PEPUn	0a.1	W	1	PE pull-up, 0=Enable 1:Disable
PDPUn	0a.0	W	1	PD pull-up, 0=Enable 1:Disable

Name	Address	R/W	Rst	Description
WKTPSC				
CPUCLKO	0b.7	W	0	1: CPUCLK output to PB2 pin 0: PB2 is general I/O if PWM1E=0 Note: CPUCLKO is higher priority than PWM1E
PWM0PE	0b.6	W	0	1: PWM0 positive output to PB1 pin 0: PB1 is general I/O
PWM0NE	0b.5	W	0	1: PWM0 negative output to PB0 pin 0: PB0 is general I/O
PWM1E	0b.4	W	0	1: PWM1 output to PB2 pin 0: PB2 is general I/O pin if CPUCLKO=0
SYNCLKO	0b.3	W	0	1: Instruction Clock (CPUCLK/2) outputs to PA3 if PAE[3]=1 and not in Fast XTAL Mode
T1OUT	0b.2	W	0	1: T1OUT to PB3 pin 0: PB3 as general I/O
WKTPSC	0b.1~0	W	11	WDT/WKT period, at V_{CC} = +5V 00 = 3.45 ms, 01 = 6.68 ms, 10 = 27.6 ms, 11 = 112 ms
ADCPSC, TM1PSC, PWM0PSC				
ADCLKS	0c.6~4	W	0	ADC clock frequency selection: 000: F_{CPUCLK}/256 001: F_{CPUCLK} /128 010: F_{CPUCLK} /64 011: F_{CPUCLK} /32 100: F_{CPUCLK} /16 101: F_{CPUCLK} /8 110: F_{CPUCLK} /4 111: F_{CPUCLK} /2
PWM0PSC	0c.3~2	W	0	PWM0 Pre-Scale, 00: F_{CPUCLK} 01: F_{CPUCLK} /2 10: F_{CPUCLK} /4 11: F_{CPUCLK} /8
T1CAPTURE	0c.1	W	0	1= Timer1 working in capture mode, Timer1 measure CAPT period (time between successive rising or falling edges of CAPT pin)
TM1PSC	0c.0	W	0	Timer1 Clock Select, 1: F_{CPUCLK}, 0: Instruction cycle (F_{CPUCLK} /2)
TM2CTRL				
INT6EDGE	0d.6	W	0	0: INT6 pin falling interrupt; 1: INT6 pin rising interrupt
INT0EDGE	0d.5	W	0	0: INT0 pin falling interrupt; 1: INT0 pin rising interrupt
TIMER2CLK	0d.4	W	0	0: Timer2 Clock is Slow Clock; 1: Timer2 Clock is F_{CPUCLK} /128
TIMER2DIV	0d.3~2	W	0	Timer2 Interrupt is Timer2 clock divided by 00: 32768 01: 16384 10: 8192 11: 128
LCDCLK	0d.1~0	W	0	LCDCLK is Timer2 clock divided by 00: 128 01: 64 10: 32 11: 16

Name	Address	R/W	Rst	Description
TKCTRL				
TKE	0e.6	W	0	1: Touch Key Enable 0: Touch Key Disable
TKSPEED	0e.5~4	W	0	Touch Key Oscillation Frequency Select 00: Fastest Touch Key clock 11: Slowest Touch Key clock
TKSEL	0e.3~0	W	0	Touch Key Channel Select 0000: TK0 0001: TK1 ……… 1111: TK15
TESTREG	0f.1~0	W	0	Test reg
CLKCTRL				
ATOSAVE	10.4	W	0	Auto Save W register and STATUS register when interrupt, and restore them when exit from interrupt.
STOPFCK	10.3	W	0	1: Stop Fast XTAL/FXRC oscillator, 0: Enable Fast XTAL/FXRC oscillator
SUBCKE	10.2	W	0	1: Enable Slow XTAL/SXRC oscillator, 0: Stop Slow XTAL/SXRC oscillator
SUBXRC	10.1	W	0	1: Slow Clock is SXRC, 0: Slow Clock is Slow XTAL
SELSUB	10.0	W	0	1: Select Slow Clock as CPUCLK, 0: Fast Clock as CPUCLK
LCDCTRL				
LCDON	11.7	W	0	1: LCD Enable 0: Disable
LCDDUTY	11.6~5	W	11	LCD Duty 00: Static 01: 1/3 duty 10: 1/4 duty 11: 1/8 duty
LCDBIAS	11.4~3	W	11	LCD Bias 00: 1/2 Bias 01: 1/3 Bias 1x: 1/4 Bias
LCDBRIT	11.2~0	W	111	LCD Brightness 000: Most darkness 111: Most brightness
PDE	12.7~0	W	0	PortD push-pull enables. Each bit controls its corresponding pin, if the bit is 0: the pin is open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output
PEE	13.7~0	W	0	PortE push-pull enables. Each bit controls its corresponding pin, if the bit is 0: the pin is open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output
PFE	14.7~0	W	0	PortF push-pull enables. Each bit controls its corresponding pin, if the bit is 0: the pin is open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output

Name	Address	R/W	Rst	Description
PGE	15.5~2	W	0	PortG push-pull enables. Each bit controls its corresponding pin, if the bit is 0: the pin is open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output
	15.1~0	W	0	PortG push-pull enables. Each bit controls its corresponding pin, if the bit is 0: the pin is pseudo-open-drain output or Schmitt-trigger input 1: the pin is CMOS push-pull output
ADPIN	16.5~0	W	3f	0=select ADC pin type, 1=select I/O pin type Each bit controls its corresponding ADC channel. Ex: bit 0: AD0; bit 1: AD1; bit 5: AD5
UBAUD	17.7~0	W	0	UART baud rate divider
SPIBR				
SPPR	18.6~4	W	0	SPI Baud Rate Prescaler Divisor 000: Divided by 1 001: Divided by 2 010: Divided by 3 011: Divided by 4 100: Divided by 5 101: Divided by 6 110: Divided by 7 111: Divided by 8
SPR	18.2~0	W	0	SPI Baud Rate Divisor 000: Divided by 2 001: Divided by 4 010: Divided by 8 011: Divided by 16 100: Divided by 32 101: Divided by 64 110: Divided by 128 111: Divided by 256
SPICR				
MSTR	19.5	W	1	1: SPI as Master, 0: SPI as Slave
CPOL	19.4	W	0	Clock Polarity, 1:Active-Low SPI clock (idles high), 0:Active-high SPI clock (idles low)
CPHA	19.3	W	0	Clock Phase, 1: first edge of SCK occurs at the begin of data 0: first edge of SCK occurs at the middle of data
SSOE	19.2	W	1	SS output enable. 1: enable, 0: disable
LSBFE	19.1	W	0	LSB first enable. 1: LSB first, 0: MSB first
SPE	19.0	W	0	1: SPI enable. 0: SPI disable
IVC_REG	1a.1~0	W	0	Regulator Control in Stop Mode 00: Vcc > 4.5V 01: 4.5V > Vcc > 3.6V 10: 3.6V > Vcc
LCDRAM	20~33	W/R	xx	LCD RAM. Initial values are not defined.
	34~3f	-	-	Not used. Read as 0x00.
SRAM	40~ff	W/R	-	192 bytes SRAM

Instruction Set

Each instruction is a 14-bit word divided into an OPCODE, which specifies the instruction type, and one or more operands, which further specify the operation of the instruction. The instructions can be categorized as byte-oriented, bit-oriented and literal operations listed in the following table.

For byte-oriented instructions, “f” or “r” represents the address designator and “d” represents the destination designator. The address designator is used to specify which address in Program memory is to be used by the instruction. The destination designator specifies where the result of the operation is to be placed. If “d” is “0”, the result is placed in the W register. If “d” is “1”, the result is placed in the address specified in the instruction.

For bit-oriented instructions, “b” represents a bit field designator, which selects the number of the bit affected by the operation, while “f” represents the address designator. For literal operations, “k” represents the literal or constant value.

Field / Legend	Description
f	F-Plane Register File Address
r	R-Plane Register File Address
b	Bit address
k	Literal. Constant data or label
d	Destination selection field, 0: Working register, 1: Register file
W	Working Register
Z	Zero Flag
C	Carry Flag
DC	Decimal Carry Flag
PC	Program Counter
TOS	Top Of Stack
GIE	Global Interrupt Enable Flag (i-Flag)
[]	Option Field
( )	Contents
.	Bit Field
B	Before
A	After
←	Assign direction

Mnemonic	Op Code	Cycle	Flag Affect	Description
Byte-Oriented File Register Instruction				
ADDWF f,d	00 0111 dfff ffff	1	C, DC, Z	Add W to f
ANDWF f,d	00 0101 dfff ffff	1	Z	AND W to f
CLRF f	00 0001 1fff ffff	1	Z	Clear f
CLRW	00 0001 0100 0000	1	Z	Clear W
COMF f,d	00 1001 dfff ffff	1	Z	Invert F bit by bit
DECF f,d	00 0011 dfff ffff	1	Z	Decrement of f
DECFSZ f,d	00 1011 dfff ffff	1 or 2	-	Decrease f, skip if zero
INCF f,d	00 1010 dfff ffff	1	Z	Increment of f
INCFSZ f,d	00 1111 dfff ffff	1 or 2	-	Increase f, skip if zero
IORWF f,d	00 0100 dfff ffff	1	Z	OR W to f
MOVFW f	00 1000 0fff ffff	1	-	Move f to W
MOVWF f	00 0000 1fff ffff	1	-	Move W to f
MOVRW r	01 1111 rrrr rrrr	1	-	Move r to W
MOVWR r	01 1110 rrrr rrrr	1	-	Move W to r
RLF f,d	00 1101 dfff ffff	1	C	F rotate to left
RRF f,d	00 1100 dfff ffff	1	C	F rotate to right
SUBWF f,d	00 0010 dfff ffff	1	C, DC, Z	Subtract W from f
SWAPF f,d	00 1110 dfff ffff	1	-	Swap high and low nibble of f
TESTZ f	00 1000 1fff ffff	1	Z	Test f if zero
XORWF f,d	00 0110 dfff ffff	1	Z	XOR W to f
Bit-Oriented File Register Instruction				
BCF f,b	01 000b bbff ffff	1	-	Bit clear f
BSF f,b	01 001b bbff ffff	1	-	Bit set f
BTFSC f,b	01 010b bbff ffff	1 or 2	-	Bit test f, skip if clear
BTFSS f,b	01 011b bbff ffff	1 or 2	-	Bit test f, skip if set
Literal and Control Instruction				
ADDLW k	01 1100 kkkk kkkk	1	C, DC, Z	Add literal to W
ANDLW k	01 1011 kkkk kkkk	1	Z	AND literal to W
XORLW k	01 1101 kkkk kkkk	1	Z	XOR literal to W
CALL k	10 kkkk kkkk kkkk	2	-	Subroutine call
CLRWDT	01 1110 0000 0100	1	TO, PD	Clear watchdog timer
GOTO k	11 kkkk kkkk kkkk	2	-	Unconditional branch
IORLW k	01 1010 kkkk kkkk	1	Z	OR literal to W
MOVLW k	01 1001 kkkk kkkk	1	-	Move literal to W
NOP	00 0000 0000 0000	1	-	No operation
RET	00 0000 0100 0000	2	-	Return from CALL
RETI	00 0000 0110 0000	2	-	Return from interrupt
RETLW k	01 1000 kkkk kkkk	2	-	Return with literal to W
SLEEP	01 1110 0000 0011	1	TO, PD	Power down

ADDLW Add Literal “k” and W

Syntax	ADDLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← (W) + k	
Status Affected	C, DC, Z	
OP-Code	01 1100 kkkk kkkk	
Description	The contents of the W register are added to the eight-bit literal ‘k’ and the result is placed in the W register.	
Cycle	1	
Example	ADDLW 0x15	B : W = 0x10 A : W = 0x25

ADDWF Add W and “f”

Syntax	ADDWF f [,d]	
Operands	f : 00h ~ 7Fh d : 0, 1	
Operation	(Destination) ← (W) + (f)	
Status Affected	C, DC, Z	
OP-Code	00 0111 dfff ffff	
Description	Add the contents of the W register with register ‘f’. If ‘d’ is 0, the result is stored in the W register. If ‘d’ is 1, the result is stored back in register ‘f’.	
Cycle	1	
Example	ADDWF FSR, 0	B : W = 0x17, FSR = 0xC2 A : W = 0xD9, FSR = 0xC2

ANDLW Logical AND Literal "k" with W

Syntax	ANDLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← (W) ‘AND’ k	
Status Affected	Z	
OP-Code	01 1011 kkkk kkkk	
Description	The contents of W register are AND’ed with the eight-bit literal 'k'. The result is placed in the W register.	
Cycle	1	
Example	ANDLW 0x5F	B : W = 0xA3 A : W = 0x03

ANDWF AND W with “f”

Syntax	ANDWF f [,d]	
Operands	f : 00h ~ 7Fh d : 0, 1	
Operation	(Destination) ← (W) ‘AND’ (f)	
Status Affected	Z	
OP-Code	00 0101 dfff ffff	
Description	AND the W register with register ‘f’. If ‘d’ is 0, the result is stored in the W register. If ‘d’ is 1, the result is stored back in register ‘f’.	
Cycle	1	
Example	ANDWF FSR, 1	B : W = 0x17, FSR = 0xC2 A : W = 0x17, FSR = 0x02

BCF Clear "b" bit of "f"

Syntax	BCF f [,b]	
Operands	f : 00h ~ 3Fh b : 0 ~ 7	
Operation	(f.b) ← 0	
Status Affected	-	
OP-Code	01 000b bbff ffff	
Description	Bit 'b' in register 'f' is cleared.	
Cycle	1	
Example	BCF FLAG_REG, 7	B : FLAG_REG = 0xC7 A : FLAG_REG = 0x47

BSF Set "b" bit of "f"

Syntax	BSF f [,b]	
Operands	f : 00h ~ 3Fh b : 0 ~ 7	
Operation	(f.b) ← 1	
Status Affected	-	
OP-Code	01 001b bbff ffff	
Description	Bit 'b' in register 'f' is set.	
Cycle	1	
Example	BSF FLAG_REG, 7	B : FLAG_REG = 0x0A A : FLAG_REG = 0x8A

BTFSC Test "b" bit of "f", skip if clear(0)

Syntax	BTFSC f [,b]	
Operands	f : 00h ~ 3Fh b : 0 ~ 7	
Operation	Skip next instruction if (f.b) = 0	
Status Affected	-	
OP-Code	01 010b bbff ffff	
Description	If bit 'b' in register 'f' is '1', then the next instruction is executed. If bit 'b' in register 'f' is '0', then the next instruction is discarded, and a NOP is executed instead, making this a 2nd cycle instruction.	
Cycle	1 or 2	
Example	LABEL1 BTFSC FLAG, 1 TRUE GOTO SUB1 FALSE ...	B : PC = LABEL1 A : if FLAG.1 = 0, PC = FALSE if FLAG.1 = 1, PC = TRUE

BTFSS Test "b" bit of "f", skip if set(1)

Syntax	BTFSS f [,b]	
Operands	f : 00h ~ 3Fh b : 0 ~ 7	
Operation	Skip next instruction if (f.b) = 1	
Status Affected	-	
OP-Code	01 011b bbff ffff	
Description	If bit 'b' in register 'f' is '0', then the next instruction is executed. If bit 'b' in register 'f' is '1', then the next instruction is discarded, and a NOP is executed instead, making this a 2nd cycle instruction.	
Cycle		
Example	LABEL1 BTFSS FLAG, 1 TRUE GOTO SUB1 FALSE ...	B : PC = LABEL1 A : if FLAG.1 = 0, PC = TRUE if FLAG.1 = 1, PC = FALSE

CALL — Call subroutine "k"

Syntax	CALL k	
Operands	K : 00h ~ FFFh	
Operation	Operation: TOS ← (PC)+ 1, PC.11~0 ← k	
Status Affected	-	
OP-Code	10 kkkk kkkk kkkk	
Description	Call Subroutine. First, return address (PC+1) is pushed onto the stack. The 12-bit immediate address is loaded into PC bits <11:0>. CALL is a two-cycle instruction.	
Cycle	2	
Example	LABEL1 CALL SUB1	B : PC = LABEL1
		A : PC = SUB1, TOS = LABEL1+1

CLRF — Clear “f”

Syntax	CLRF f	
Operands	f : 00h ~ 7Fh	
Operation	(f) ← 00h, Z ← 1	
Status Affected	Z	
OP-Code	00 0001 1fff ffff	
Description	The contents of register ‘f’ are cleared and the Z bit is set.	
Cycle	1	
Example	CLRF FLAG_REG	B : FLAG_REG = 0x5A
		A : FLAG_REG = 0x00, Z = 1

CLRW — Clear W

Syntax	CLRW	
Operands	-	
Operation	(W) ← 00h, Z ← 1	
Status Affected	Z	
OP-Code	00 0001 0100 0000	
Description	W register is cleared and Zero bit (Z) is set.	
Cycle	1	
Example	CLRW	B : W = 0x5A
		A : W = 0x00, Z = 1

CLRWDT — Clear Watchdog Timer

Syntax	CLRWDT	
Operands	-	
Operation	WDTE ← 00h	
Status Affected	TO,PD	
OP-Code	00 0000 0000 0100	
Description	CLRWDT instruction enables and resets the Watchdog Timer.	
Cycle	1	
Example	CLRWDT	B : WDT counter = ?
		A : WDT counter = 0x00

COMF	Complement “f”	
Syntax	COMF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	(destination) ← ($\overline{f}$)	
Status Affected	Z	
OP-Code	00 1001 dfff ffff	
Description	The contents of register ‘f’ are complemented. If ‘d’ is 0, the result is stored in W. If ‘d’ is 1, the result is stored back in register ‘f’.	
Cycle	1	
Example	COMF REG1,0	B : REG1 = 0x13 A : REG1 = 0x13, W = 0xEC

DECF	Decrement “f”	
Syntax	DECF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	(destination) ← (f) - 1	
Status Affected	Z	
OP-Code	00 0011 dfff ffff	
Description	Decrement register ‘f’. If ‘d’ is 0, the result is stored in the W register. If ‘d’ is 1, the result is stored back in register ‘f’.	
Cycle	1	
Example	DECF CNT, 1	B : CNT = 0x01, Z = 0 A : CNT = 0x00, Z = 1

DECFSZ	Decrement “f”, Skip if 0	
Syntax	DECFSZ f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	(destination) ← (f) - 1, skip next instruction if result is 0	
Status Affected	-	
OP-Code	00 1011 dfff ffff	
Description	The contents of register ‘f’ are decremented. If ‘d’ is 0, the result is placed in the W register. If ‘d’ is 1, the result is placed back in register ‘f’. If the result is 1, the next instruction is executed. If the result is 0, then a NOP is executed instead, making it a 2 cycle instruction.	
Cycle	1 or 2	

Example	LABEL1 DECFSZ CNT, 1
	GOTO LOOP
	CONTINUE
		B : PC = LABEL1
		A : CNT = CNT – 1
		if CNT=0, PC = CONTINUE
		if CNT≠0, PC = LABEL1+1

GOTO	Unconditional Branch	
Syntax	GOTO k	
Operands	k : 00h ~ FFFh	
Operation	PC.11~0 ← k	
Status Affected	-	
OP-Code	11 kkkk kkkk kkkk	
Description	GOTO is an unconditional branch. The 12-bit immediate value is loaded into PC bits <11:0>. GOTO is a two-cycle instruction.	
Cycle	2	
Example	LABEL1 GOTO SUB1	B : PC = LABEL1 A : PC = SUB1

INCF Increment “f”

Syntax	INCF f [,d]	
Operands	f : 00h ~ 7Fh	
Operation	(destination) ← (f) + 1	
Status Affected	Z	
OP-Code	00 1010 dfff ffff	
Description	The contents of register ‘f’ are incremented. If ‘d’ is 0, the result is placed in the W register. If ‘d’ is 1, the result is placed back in register ‘f’.	
Cycle	1	
Example	INCF CNT, 1	B : CNT = 0xFF, Z = 0 A : CNT = 0x00, Z = 1

INCFSZ Increment “f”, Skip if 0

Syntax	INCFSZ f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	(destination) ← (f) + 1, skip next instruction if result is 0	
Status Affected	-	
OP-Code	00 1111 dfff ffff	
Description	The contents of register ‘f’ are incremented. If ‘d’ is 0, the result is placed in the W register. If ‘d’ is 1, the result is placed back in register ‘f’. If the result is 1, the next instruction is executed. If the result is 0, a NOP is executed instead, making it a 2 cycle instruction.	
Cycle	1 or 2	

Example	LABEL1 INCFSZ CNT, 1
	GOTO LOOP
	CONTINUE
		B : PC = LABEL1
		A : CNT = CNT + 1
		if CNT=0, PC = CONTINUE
		if CNT≠0, PC = LABEL1+1

IORLW Inclusive OR Literal with W

Syntax	IORLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← (W) OR k	
Status Affected	Z	
OP-Code	01 1010 kkkk kkkk	
Description	The contents of the W register are OR’ed with the eight-bit literal 'k'. The result is placed in the W register.	
Cycle	1	
Example	IORLW 0x35	B : W = 0x9A A : W = 0xBF, Z = 0

IORWF Inclusive OR W with “f”

Syntax	IORWF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	(destination) ← (W) OR (f)	
Status Affected	Z	
OP-Code	00 0100 dfff ffff	
Description	Inclusive OR the W register with register ‘f’. If ‘d’ is 0, the result is placed in the W register. If ‘d’ is 1, the result is placed back in register ‘f’.	
Cycle	1	
Example	IORWF RESULT, 0	B : RESULT = 0x13, W = 0x91 A : RESULT = 0x13, W = 0x93, Z = 0

MOVFW Move “f” to W

Syntax	MOVFW f	
Operands	f : 00h ~ 7Fh	
Operation	(W) ← (f)	
Status Affected	-	
OP-Code	00 1000 0fff ffff	
Description	The contents of register f are moved to W register.	
Cycle	1	
Example	MOVF FSR, 0	B : W = ? A : W ← f, if W = 0 Z = 1

MOVLW Move Literal to W

Syntax	MOVLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← k	
Status Affected	-	
OP-Code	01 1001 kkkk kkkk	
Description	The eight-bit literal ‘k’ is loaded into W register. The don’t cares will assemble as 0’s.	
Cycle	1	
Example	MOVLW 0x5A	B : W = ? A : W = 0x5A

MOVWF Move W to “f”

Syntax	MOVWF f	
Operands	f : 00h ~ 7Fh	
Operation	(f) ← (W)	
Status Affected	-	
OP-Code	00 0000 1fff ffff	
Description	Move data from W register to register ‘f’.	
Cycle	1	
Example	MOVWF REG1	B : REG1 = 0xFF, W = 0x4F A : REG1 = 0x4F, W = 0x4F

MOVWR Move W to “r”

Syntax	MOVWR r	
Operands	r : 00h ~ FFh	
Operation	(r) ← (W)	
Status Affected	-	
OP-Code	01 1110 rrrr rrrr	
Description	Move data from W register to register ‘r’.	
Cycle	1	
Example	MOVWR REG1	B : REG1 = 0xFF, W = 0x4F A : REG1 = 0x4F, W = 0x4F

MOVRW Move “r” to W

Syntax	MOVRW r	
Operands	r : 20h ~ FFh	
Operation	(W) ← (r)	
Status Affected	-	
OP-Code	01 1111 rrrr rrrr	
Description	Move data from register ‘r’ to W register.	
Cycle	1	
Example	MOVRW REG1	B : REG1 = 0x4F, W = ? A : REG1 = 0x4F, W = 0x4F

NOP No Operation

Syntax	NOP	
Operands	-	
Operation	No Operation	
Status Affected	-	
OP-Code	00 0000 0000 0000	
Description	No Operation	
Cycle	1	
Example	NOP	-

RETI Return from Interrupt

Syntax	RETI	
Operands	-	
Operation	PC ← TOS, GIE ← 1	
Status Affected	-	
OP-Code	00 0000 0110 0000	
Description	Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in to the PC. Interrupts are enabled. This is a two-cycle instruction.	
Cycle	2	
Example	RETFIE	A : PC = TOS, GIE = 1

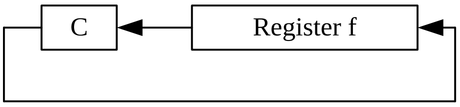
RETLW Return with Literal in W

Syntax	RETLW k	
Operands	k : 00h ~ FFh	
Operation	PC ← TOS, (W) ← k	
Status Affected	-	
OP-Code	01 1000 kkkk kkkk	
Description	The W register is loaded with the 8-bit literal ‘k’. The program counter is loaded from the top of the stack (the return address). This is a two-cycle instruction.	
Cycle	2	
Example	CALL TABLE : TABLE ADDWF PCL,1 RETLW k1 RETLW k2 : RETLW kn	B : W = 0x07 A : W = value of k8

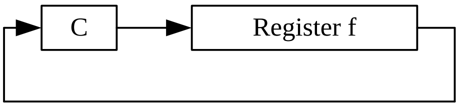
RET Return from Subroutine

Syntax	RET	
Operands	-	
Operation	PC ← TOS	
Status Affected	-	
OP-Code	00 0000 0100 0000	
Description	Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a two-cycle instruction.	
Cycle	2	
Example	RETURN	A : PC = TOS

RLF Rotate Left f through Carry

Syntax	RLF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation		
Status Affected	C	
OP-Code	00 1101 dfff ffff	
Description	The contents of register ‘f’ are rotated one bit to the left through the Carry Flag. If ‘d’ is 0, the result is placed in the W register. If ‘d’ is 1, the result is stored back in register ‘f’.	
Cycle	1	
Example	RLF REG1,0	B : REG1 = 1110 0110, C = 0 A : REG1 = 1110 0110 W = 1100 1100, C = 1

RRF Rotate Right “f” through Carry

Syntax	RRF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation		
Status Affected	C	
OP-Code	00 1100 dfff ffff	
Description	The contents of register ‘f’ are rotated one bit to the right through the Carry Flag. If ‘d’ is 0, the result is placed in the W register. If ‘d’ is 1, the result is placed back in register ‘f’.	
Cycle	1	
Example	RRF REG1,0	B : REG1 = 1110 0110, C = 0 A : REG1 = 1110 0110 W = 0111 0011, C = 0

SLEEP Go into standby mode, Clock oscillation stops

Syntax	SLEEP	
Operands	-	
Operation	-	
Status Affected	TO,PD	
OP-Code	00 0000 0000 0011	
Description	Go into SLEEP mode with the oscillator stopped.	
Cycle	1	
Example	SLEEP	-

SUBWF Subtract W from “f”

Syntax	SUBWF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	(W) ← (f) – (W)	
Status Affected	C, DC, Z	
OP-Code	00 0010 dfff ffff	
Description	Subtract (2’s complement method) W register from register ‘f’. If ‘d’ is 0, the result is stored in the W register. If ‘d’ is 1, the result is stored back in register ‘f’.	
Cycle	1	
Example	SUBWF REG1,1	B : REG1 = 3, W = 2, C = ?, Z = ? A : REG1 = 1, W = 2, C = 1, Z = 0
	SUBWF REG1,1	B : REG1 = 2, W = 2, C = ?, Z = ? A : REG1 = 0, W = 2, C = 1, Z = 1
	SUBWF REG1,1	B : REG1 = 1, W = 2, C = ?, Z = ? A : REG1 = FFh, W = 2, C = 0, Z = 0

SWAPF Swap Nibbles in “f”

Syntax	SWAPF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	(destination,7~4) ← (f.3~0), (destination.3~0) ← (f.7~4)	
Status Affected	-	
OP-Code	00 1110 dfff ffff	
Description	The upper and lower nibbles of register ‘f’ are exchanged. If ‘d’ is 0, the result is placed in W register. If ‘d’ is 1, the result is placed in register ‘f’.	
Cycle	1	
Example	SWAPF REG, 0	B : REG1 = 0xA5 A : REG1 = 0xA5, W = 0x5A

TESTZ Test if “f” is zero

Syntax	TESTZ f	
Operands	f : 00h ~ 7Fh	
Operation	Set Z flag if (f) is 0	
Status Affected	Z	
OP-Code	00 1000 1fff ffff	
Description	If the content of register ‘f’ is 0, Zero flag is set to 1.	
Cycle	1	
Example	TESTZ REG1	B : REG1 = 0, Z = ? A : REG1 = 0, Z = 1

XORLW Exclusive OR Literal with W

Syntax	XORLW k	
Operands	k : 00h ~ FFh	
Operation	(W) ← (W) XOR k	
Status Affected	Z	
OP-Code	01 1111 kkkk kkkk	
Description	The contents of the W register are XOR’ed with the eight-bit literal ‘k’. The result is placed in the W register.	
Cycle	1	
Example	XORLW 0xAF	B : W = 0xB5 A : W = 0x1A

XORWF Exclusive OR W with “f”

Syntax	XORWF f [,d]	
Operands	f : 00h ~ 7Fh, d : 0, 1	
Operation	(destination) ← (W) XOR (f)	
Status Affected	Z	
OP-Code	00 0110 dfff ffff	
Description	Exclusive OR the contents of the W register with register ‘f’. If ‘d’ is 0, the result is stored in the W register. If ‘d’ is 1, the result is stored back in register ‘f’.	
Cycle	1	
Example	XORWF REG 1	B : REG = 0xAF, W = 0xB5 A : REG = 0x1A, W = 0xB5

Electrical Characteristics

1. Absolute Maximum Ratings (T$_A$ = 25℃)

Parameter	Rating	Unit
Supply voltage	V_{SS} - 0.3 to V_{SS} + 5.5	V
Input voltage	V_{SS} - 0.3 to V_{CC} + 0.3	
Output voltage	V_{SS} - 0.3 to V_{CC} + 0.3	
Output current high per 1 PIN	-25	mA
Output current high per all PIN	-80	
Output current low per 1 PIN	+30	
Output current low per all PIN	+150	
Maximum Operating Voltage (TM57FLA80)	5.5	V
Maximum Operating Voltage (TM57FLA80A)	3.5	V
Operating temperature	-40 to +85	℃
Storage temperature	-65 to +150	

2. DC Characteristics (T_A = -25℃ to +85℃, Vcc1 = 2.0V to 5.5V for TM57FLA80 ; Vcc = 2.0V to 3.3V for TM57FLA80A)

Parameter	Symbol	Conditions			Min	Typ	Max	Unit
Input High Voltage	V_{IH}	All Input	V_{CC} = 2.0 to Vcc1		0.8 V_{CC}	–	V_{CC}	V
Input Low Voltage	V_{IL}	All Input	V_{CC} = 2.0 to Vcc1		0	–	0.2 V_{CC}	V
Output High Voltage	V_{OH}	All Output	V_{CC} =5V, I_{OH}= -13mA		Vcc - 0.7	–	–	V
			V_{CC}=3V, I_{OH}=-7.5 mA					
Output Low Voltage	V_{OL}	All Output	V_{CC} = 5V, I_{OL}=20 mA		–	–	0.5	V
			V_{CC} = 3V, I_{OL}=10 mA					
Input Leakage Current (pin high)	I_{ILH}	All Input	V_{IN} = V_{CC}		–	–	1	µA
Input Leakage Current (pin low)	I_{ILL}	All Input	V_{IN} = 0 V		–	–	–1	µA
Output Leakage Current (pin high)	I_{OLH}	All Output	V_{OUT} = V_{CC}		–	–	2	µA
Output Leakage Current (pin low)	I_{OLL}	All Output	V_{OUT} = 0 V		–	–	–2	µA
Power Supply Current	I_{DD}	Run 12 MHz	V_{CC} = 3V		-	4.0	-	mA
			V_{CC} = 5V			6.5		
		Run 4 MHz	V_{CC} = 3V		-	1.5	-	
			V_{CC} = 5V			3.6		
		Idle Mode (32K enable)	V_{CC}=3V	LCD ON		17	25	µA
				LCD OFF		2	4	
			V_{CC}=5V	LCD ON		36	45	
				LCD OFF		11	16	
		Stop Mode	V_{CC} = 3V~5.5V (LCD OFF, IVC OFF)		–	0.1	1	µA
		Stop Mode (LCD OFF, IVC ON)	V_{CC} = 3V		-	72	85	
			V_{CC} = 5V			188	200	
		Stop Mode	V_{CC} = 3V (LCD ON, IVC OFF)			16	24	
		Slow Mode	V_{CC} = 5V	LCD ON	-	240	260	µA
				LCD OFF		215	235	µA
			V_{CC} = 3V	LCD ON	-	113	125	µA
				LCD OFF		100	112	µA
Pull-Up Resistor	R_P	V_{IN} = 0 V Ports A	V_{CC} = 5 V		25	50	100	kΩ
LCD Voltage Divider Resistor	R_{LCD}	-	-			200		kΩ

3. Clock Timing (T_A = -25℃ to +85℃, V_{CC} = 2.0V to 5.5V)

Parameter	Condition			Min	Typ	Max	Unit
External RC Frequency	V_{CC} = 3V	R = 5.1K	C = 30 pF		2.15		MHz
		R = 10K	C = 100 pF		0.34		
		R = 100K	C = 300 pF		0.04		
	V_{CC} = 5V	R = 5.1K	C = 30 pF		2.45		
		R = 10K	C = 100 pF		0.65		
		R = 100K	C = 300 pF		0.03		
Internal RC Frequency	V_{CC} = 4.75 to 5.25V			3.88	4.00	4.12	
	V_{CC} = 2.8 to 3.2V			3.90	4.02	4.10	

4. Reset Timing Characteristics (T_A = -25℃ to +85℃, V_{CC} = 2.1V to 5.5V)

Parameter	Conditions	Min	Typ	Max	Unit
Input High Voltage	–	0.8 V_{CC}	–	V_{CC}	V
Input Low Voltage	–	–	–	0.2 V_{CC}	V
RESET Input Low width	Input V_{CC} = 5V ± 10 %	3	–	–	µs
WDT wakeup time	V_{CC} = 5V, WKTPSC = 11	96	113	130	ms
	V_{CC} = 3V, WKTPSC = 11	103	121	149	
CPU start up time	System Clock = 12 MHz	–	–	100	µs

5. LVR Circuit Characteristics (T_A =-25℃ to +85℃, V_{CC} = 2.0V to 5.5V)

Parameter	Symbol	Min	Typ	Max	Unit
LVR reference Voltage	V_{LVR}	1.9 2.6	2.1 2.9	2.3 3.2	V
LVR Hysteresis Voltage	V_{HYST}	–	±0.2	–	V
Low Voltage Detection time	t_{LVR}	100	–	–	µs

Note that TM57FLA80A doesn't support LVR voltages.

6. ADC Electrical Characteristics (T_A =-25℃ to +85℃, V_{CC} = 2.0V to 5.5V)

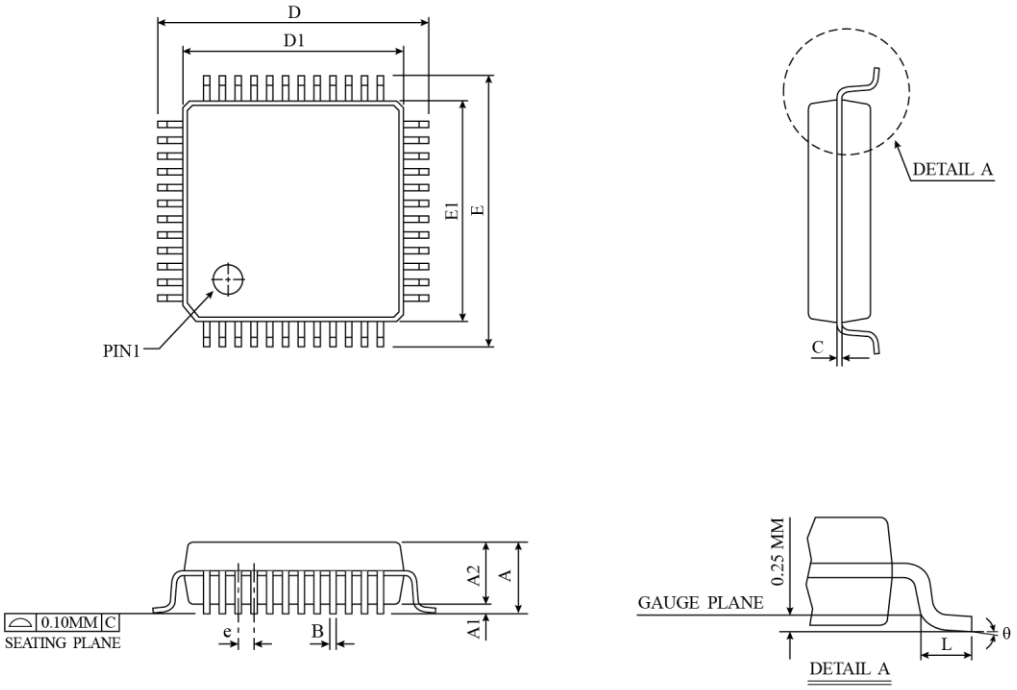
Parameter	Conditions	Min	Typ	Max	Units
Total Accuracy	V_{CC} = Vcc1, V_{SS} = 0V	–	–	± 3	LSB
Integral Non-Linearity		–	–	± 2	
Offset Error of Top		–	± 1	± 3	
Offset Error of Bottom		–	± 1	± 2	
Max Input Clock (f_{ADC})	–	–	–	2	MHz
Conversion Time	f_{ADC} = 2 MHz	–	25	–	µs
Input Voltage	–	V_{SS}	–	V_{REF}	V
Input Impedance	–	2	–	–	M
Input Current	V_{CC} = 5V	–	–	10	µA
VREF Range	–	V_{CC} -0.6	V_{CC}	V_{CC}	V

Package Information

The ordering information:

Ordering number	Package
TM57FLA80-MTP	Wafer / Dice blank chip
TM57FLA80-COD	Wafer / Dice with code
TM57FLA80-MTP-24	SOP 32-pin (300 mil)
TM57FLA80-MTP-74	QFP 44-pin
TM57FLA80-MTP-72	LQFP 48-pin
TM57FLA80A-MTP	Wafer / Dice blank chip
TM57FLA80A-COD	Wafer / Dice with code
TM57FLA80A-MTP-24	SOP 32-pin (300 mil)
TM57FLA80A-MTP-74	QFP 44-pin
TM57FLA80A-MTP-72	LQFP 48-pin

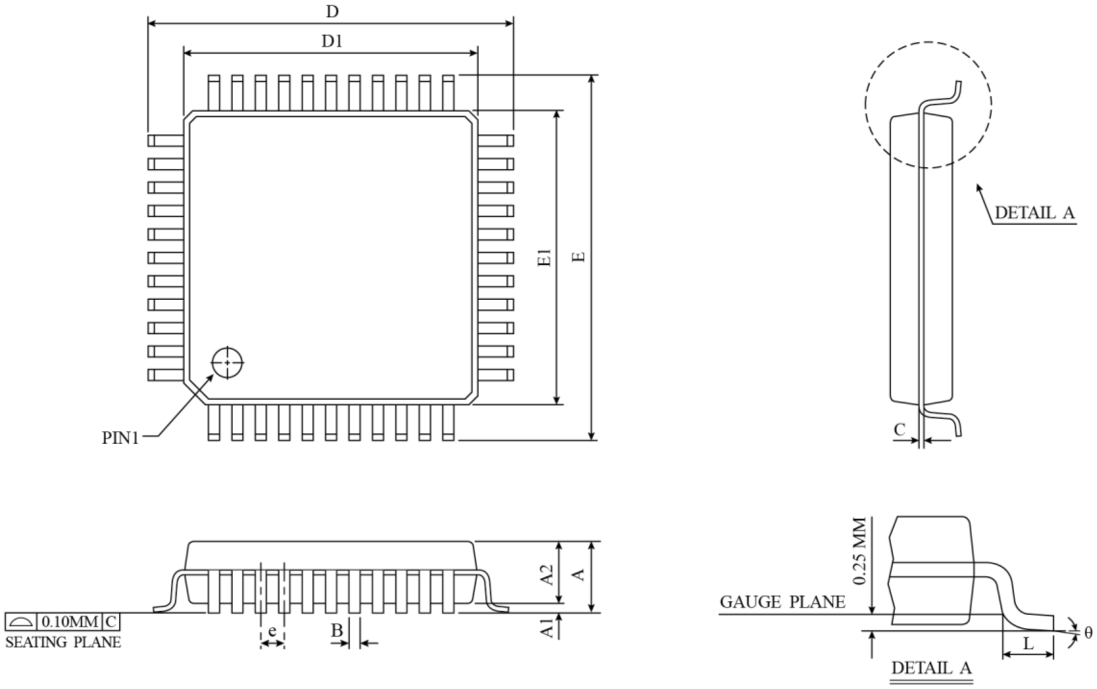
LQFP-48 Package Dimension



SYMBOL	DIMENSION IN MM		DIMENSION IN INCH	
	MIN	MAX	MIN	MAX
A	-	1.60	-	0.063
A1	0.05	0.15	0.001	0.006
A2	1.35	1.45	0.053	0.057
B	0.17	0.27	0.007	0.011
C	0.09	0.20	0.004	0.008
D	9.00 BSC		0.354 BSC	
D1	7.00 BSC		0.276 BSC	
E	9.00 BSC		0.354 BSC	
E1	7.00 BSC		0.276 BSC	
e	0.50 BSC		0.020 BSC	
L	0.45	0.75	0.018	0.030
θ	0°	7°	0°	7°
JEDEC	MS-026 (BBC)			

△ * NOTES : DIMENSION 〝D1〞 AND 〝E1〞 DO NOT INCLUDE MOLD
PROTRUSIONS. ALLOWABLE PROTRUSIONS IS 0.25mm PER SIDE.
〝D1〞 AND 〝E1〞 ARE MAXIMUM PLASTIC BODY SIZE DIMENSIONS
INCLUDING MOLD MISMACH.

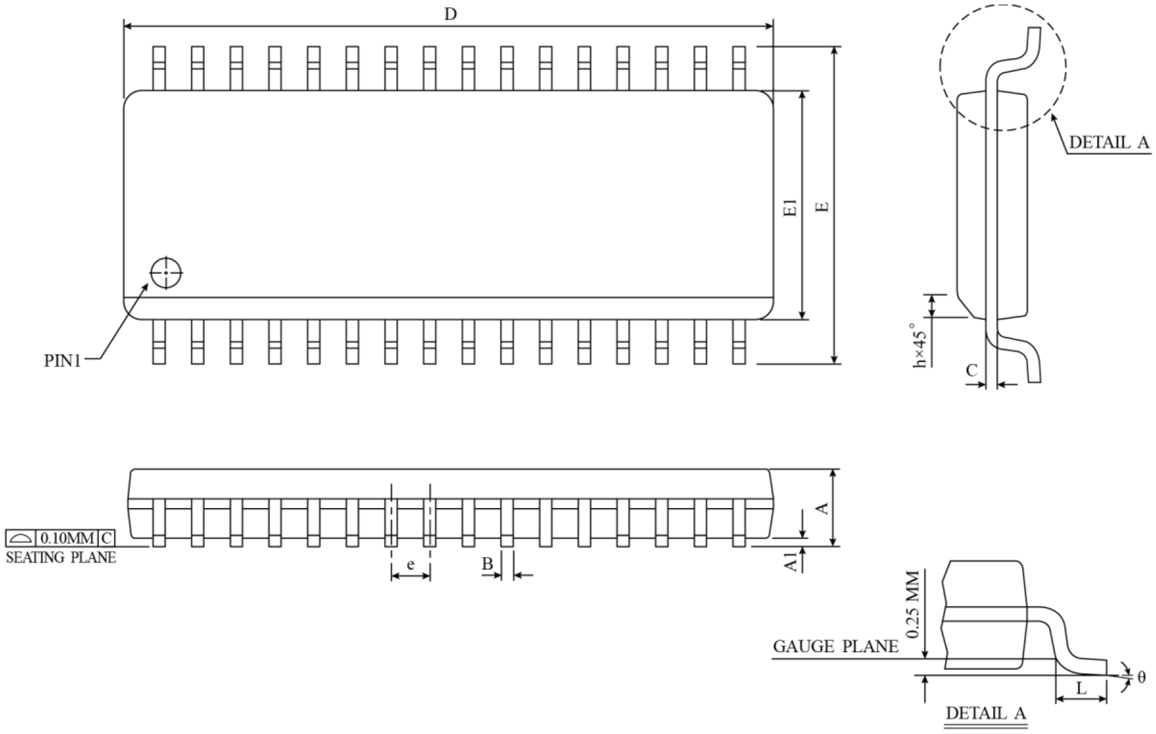
QFP-44 Package Dimension



SYMBOL	DIMENSION IN MM		DIMENSION IN INCH	
	MIN	MAX	MIN	MAX
A	1.95	2.30	0.077	0.091
A1	0.05	0.20	0.002	0.008
A2	1.90	2.10	0.075	0.083
B	0.30	0.38	0.012	0.015
C	0.11	0.23	0.004	0.009
D	12.9	13.5	0.508	0.531
D1	9.90	10.1	0.390	0.398
E	12.9	13.5	0.508	0.531
E1	9.90	10.1	0.390	0.398
e	0.67	0.93	0.026	0.037
L	0.60	1.00	0.024	0.039
θ	0°	7°	0°	7°
JEDEC				

⚠ * NOTES : BOTH PACKAGE LENGTH AND WIDTH DO NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS, PROTRUSIONS OR GATE BURRS SHALL NOT EXCEED 0.15 mm PER END.

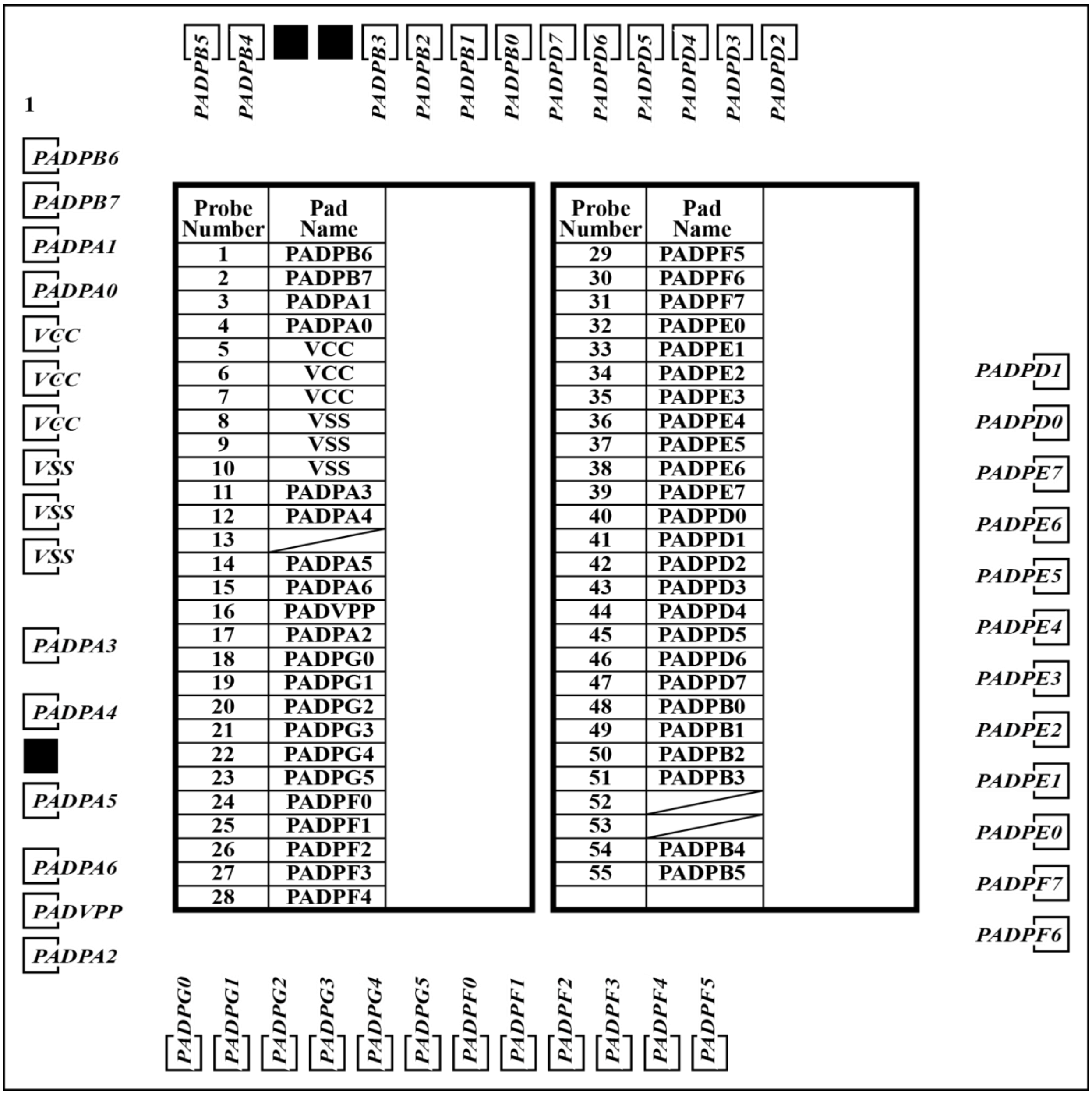
SOP-32 Package Dimension



SYMBOL	DIMENSION IN MM		DIMENSION IN INCH	
	MIN	MAX	MIN	MAX
A	2.35	2.65	0.0926	0.1043
A1	0.10	0.30	0.0040	0.0118
B	0.33	0.51	0.013	0.020
C	0.23	0.32	0.0091	0.0125
D	20.32	20.73	0.800	0.816
E	10.00	10.65	0.394	0.491
E1	7.40	7.60	0.2914	0.2992
e	1.27 BSC		0.050 BSC	
h	0.25	0.75	0.010	0.029
L	0.40	1.27	0.016	0.050
θ	0°	8°	0°	8°

△A * NOTES : DIMENSION 〝D〞 DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS.
MOLD FLASH, PROTRUSIONS AND GATE BURRS SHALL
NOT EXCEED 0.15 MM (0.006 INCH) PER SIDE.

Pads Diagram

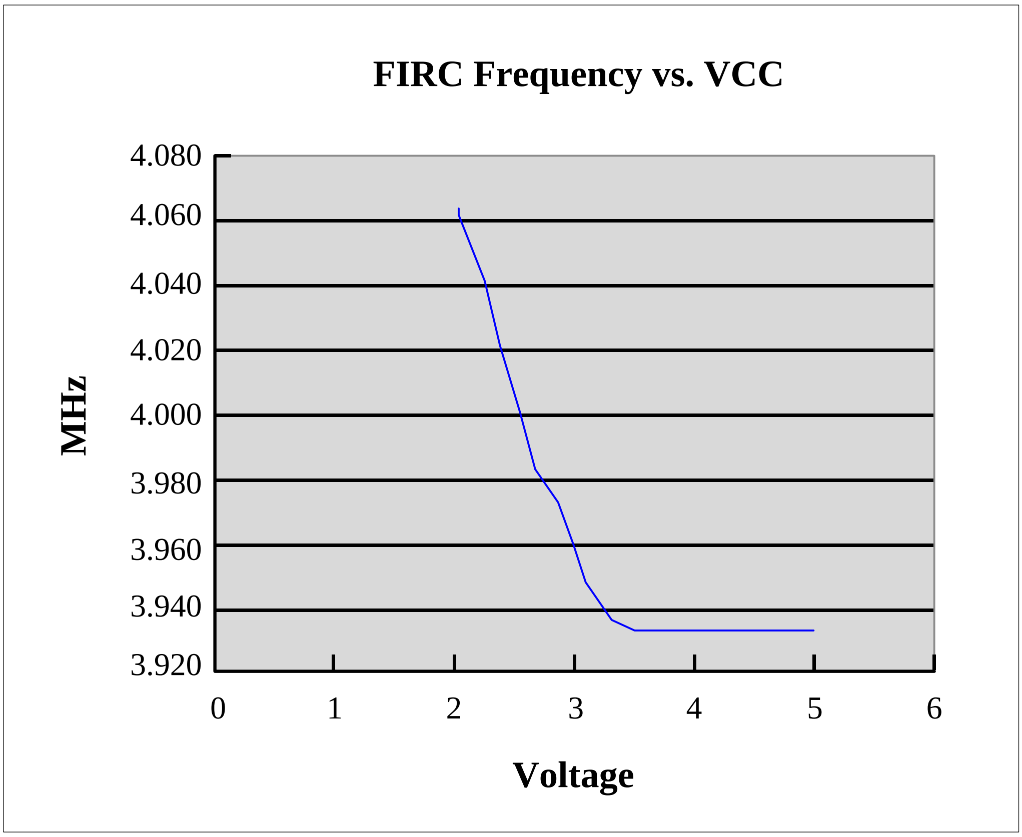


Probe Number	Pad Name	
1	PADPB6	
2	PADPB7	
3	PADPA1	
4	PADPA0	
5	VCC	
6	VCC	
7	VCC	
8	VSS	
9	VSS	
10	VSS	
11	PADPA3	
12	PADPA4	
13		
14	PADPA5	
15	PADPA6	
16	PADVPP	
17	PADPA2	
18	PADPG0	
19	PADPG1	
20	PADPG2	
21	PADPG3	
22	PADPG4	
23	PADPG5	
24	PADPF0	
25	PADPF1	
26	PADPF2	
27	PADPF3	
28	PADPF4	

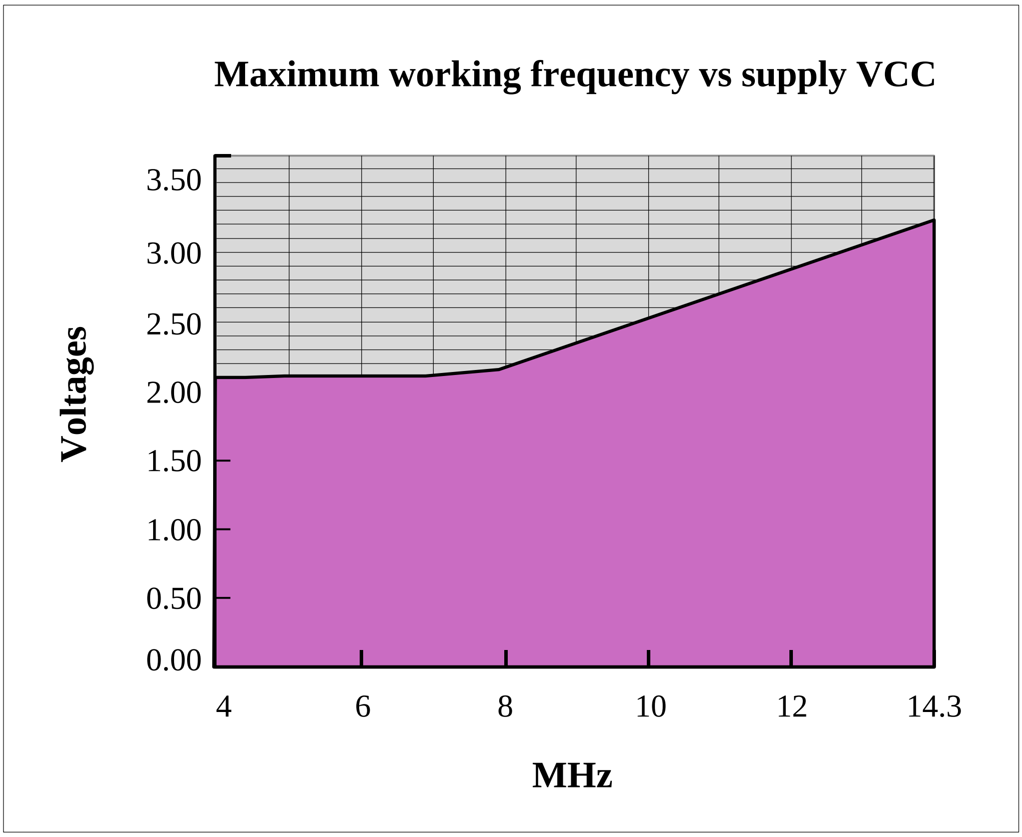
Probe Number	Pad Name	
29	PADPF5	
30	PADPF6	
31	PADPF7	
32	PADPE0	
33	PADPE1	
34	PADPE2	
35	PADPE3	
36	PADPE4	
37	PADPE5	
38	PADPE6	
39	PADPE7	
40	PADPD0	
41	PADPD1	
42	PADPD2	
43	PADPD3	
44	PADPD4	
45	PADPD5	
46	PADPD6	
47	PADPD7	
48	PADPB0	
49	PADPB1	
50	PADPB2	
51	PADPB3	
52		
53		
54	PADPB4	
55	PADPB5	

Frequency of FIRC vs. Supply Voltage V_{CC}

FIRC frequency is trimmed to 4 MHz by tenx. The frequency deviation is ±3% from lowest to highest supply voltage.



Maximum Working Frequency vs. Supply Voltage V_{CC}


Maximum working frequency vs supply VCC
Voltages
MHz