JAM PENGGERA DIGIT BERDASARKAN MICRO-PENGAWAL PIC

Oleh

Wong Echor

Disertasi ini dikemukakan kepada
UNIVERSITI SAINS MALAYSIA

Sebagai memenuhi sebahagian daripada syarat keperluan
untuk ijazah dengan kepujian

SARJANA MUDA KEJURUTERAAN (KEJURUTERAAN ELEKTRONIK)

Pusat Pengajian Kejuruteraan
Elektrik dan Elektronik
Universiti Sains Malaysia

Mac 2005

DIGITAL ALARM CLOCK BASED ON PIC MICROCONTROLLER

By

Wong Echor

Dissertation submitted to
UNIVERSITI SAINS MALAYSIA

In partial fulfillment of the requirements
for degree with honors

BACHELOR OF ENGINEERING (ELECTRONIC ENGINEERING)

School of Electronic and
Electrical Engineering
Universiti Sains Malaysia

Mac 2005

ABSTRAK

Jam penggera digit sudah wujud sekian lama dalam pasaran, tetapi kebanyakan produk yang ada terdapat banyak kelemahan. Contohnya, kebanyakan jam penggera tidak mempunyai fungsi yang menunjukkan hari minggu (Ahad hingga Sabtu) dan hanya boleh menyimpan satu masa penggera.

Tujuan projek ini ialah untuk mencadangkan satu penyelesaian kepada kelemahan jam penggera digital. Projek ini membincangkan tentang perekaan jam penggera digit yang berkos rendah dengan menggunakan mikro pengawal yang popular iaitu PIC16F84A dari Microchip.

Jam penggera digit yang direka boleh menunjukkan hari minggu iaitu dari Ahad ke Isnin dan ia boleh menyimpan lapan masa penggera. Setiap masa penggera boleh disetkan supaya bergera setiap hari, hanya hari bekerja atuapun pada hari yang spesifik sahaja mengikut kekendak pengguna. Kesemua lapan masa penggera disimpan pada EEPROM mikro pengawal PIC. Jadi, pengguna tidak perlu bimbang akan kehilangan masa pengera yang telah diset walaupun kuasa putus.

Seperti jam penggera lain yang bagus, jam penggera yang direka mempunyai persediaan bateri untuk kegunaan semasa kuasa AC gagal supaya jam penggera akan terus berfungsi. Dan penggunana penganyun kristal 4-MHz untuk memastikan masa jam adalah tepat.

ABSTRACT

Although digital alarm clock have been around for years, most of today's off the shelf products suffer from serious design limitations. For example, many do not keep track of weekdays (Sunday to Saturday) and can only store one alarm time.

The project proposes a solution for the design weaknesses of the alarm clock by using a low-cost circuit based on the popular PIC 16F84A microcontroller from Microchip.

The digital alarm clock keeps track of weekdays and has eight alarms that can be individually set to go off every day, only on working day or on a specific day of the week. The eight different alarm times are kept in the PIC's EEPROM. So, the user would not have to adjust any of the alarms again even if the battery fails.

Like all good digital alarm clocks, this project has a battery that keeps the clock ticking in the event of an AC-power failure. And the 4-MHz crystal oscillator guarantees a pretty accurate timebase for the clock.

ACKNOWLEDGMENT

A journey is easier when you travel together. Interdependence is certainly more valuable than independence. This project is the result of one year of work whereby I have been accompanied and supported by many people. It is a pleasant aspect that I have now the opportunity to express my gratitude for all of them.

Firstly, I wish to take the opportunity to thank my supervisor of my final project Pn Chean Kooi Chyi for her advises and teaching. Her extensive knowledge, enlightening direction, and continuous encouragement made my project work smooth, positive, and enjoyable. She help me in many way to make my project success. I really appreciate her help. I also want to thank all staff of the School of Electronic and Electric for their commitment and help.

I also wish to express my appreciation to Prof. Madya Mohd. Yusoff Mashor for his invaluable support and guidance throughout the course of this report.

I wish to thank my industrial training supervisor Mr Kang Joo Song, the managing director of Affinex Technology Sdn Bhd Mr Kang Joo Song for allow me to make my PCB in his laboratory.

I wish to thank my parents, who always emphasized the important of education as means of achieving a happy life and also providing me encouragement for this project. Without their caring and support, the achievements of my project are impossible.

Finally, I am thankful for my faith that keeps me progressing eternally.

TABLE OF CONTENTS

Page

LIST OF FIGURES AND TABLES

CHAPTER 1 : INTRODUCTION

1.1 Project introduction

The objective of this project is to produce a low-cost digital alarm clock that can solve the problems or weaknesses of the digital alarm clock in market. In this project, Microcontroller PIC 16F84A from Microchip is used because of its reliability and cost.

The digital alarm clock has been designed that it can keeps track of weekdays (Sunday to Saturday) and has eight alarms that can be individually set to go off every day, only on working days or, on a specific day of the week.

The time setting process of this alarm clock allows the adjustment of each digit of the clock separately by means of Up and Down (+ and -) keys. It also incorporates some other interesting features.

Like all good digital alarm clocks, in this project a backup battery is included to keeps the clock ticking in the event of an AC power failure. In battery-powered mode the display is turn off to reduce the energy consumption. However, the user can still able to check the time by pressing a 'display on' button. In battery mode the alarm will continue to operate normally. When an alarm goes off the display will be turned on to show the current time.

The eight different alarm times are kept in the PIC's EEPROM. So, the users won't have to adjust any of the alarms again even if the battery fails. A 'snooze' key will temporarily turn an alarm off. The alarm will be triggered again after a minute until it is definitely turned off. A master Alarm On/Off key will enable/disable all the alarms, independent of their individually set states.

A bright LED display makes it easy to check the time from a distance or in dark. And the 4-Mhz crystal oscillator guarantees a pretty accurate timebase for the clock (error = 0.000427%)

CHAPTER 2 : PIC MICROCONTROLLER

2.1 INTRODUCTION

PIC(Peripheral Interface Controller) is the IC which was developed to control peripheral devices, alleviating the load from the main CPU. Compared to a human being, the brain is the main CPU and the PIC is equivalent to the autonomic nervous system.

The PIC, like the CPU, has calculation functions and memory, and is controlled by the software. However, the throughput and the memory capacity are low. Depending on the kind of PIC, the maximum clock operating frequency is about 20 MHz and the memory capacity (to write the program) is about 1K to 4K words.

The PIC is convenient for making calculations. The memory, the input/output ports and so on are incorporated into the IC. The efficiency and the functions are limited, but the PIC can do the job of many IC's with software. So, the circuit can be compact. Figure 2.1 shows the PIC16F84A microcontroller.

Figure2.1 : PIC 16F84A microcontroller

2.1.1 Microcontroller PIC 16F84A

PIC16F84 belongs to a class of 8-bit microcontrollers of RISC (Reduced

Instruction Set Computer) architecture. That means that it has a reduced set of instructions, more precisely 35 instructions. All of these instructions are executed in one cycle except for jump and branch instructions. Its general structure is shown in Figure 2.2:

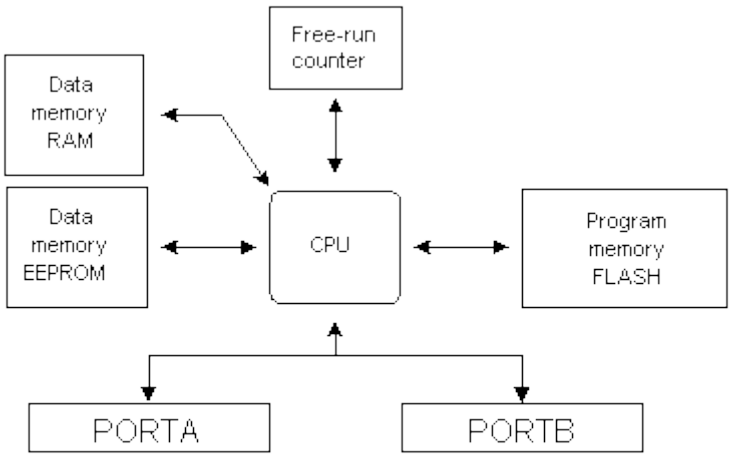


Figure 2.2 : PIC16F84A Outline

Program memory (FLASH)- for storing a written program.
Since memory made in FLASH technology can be programmed and cleared more than once, it makes this microcontroller suitable for device development.

EEPROM - data memory that needs to be saved when there is no supply.
It is usually used for storing important data that must not be lost if power supply suddenly stops. For instance, one such data is an assigned temperature in temperature regulators. If during a loss of power supply this data was lost, we would have to make the adjustment once again upon return of supply. Thus our device looses on self-reliance.

RAM - data memory used by a program during its execution.
In RAM are stored all inter-results or temporary data during run-time.

PORTA and PORTB are physical connections between the microcontroller and the

outside world. Port A has five, and port B has eight pins.

FREE-RUN TIMER is an 8-bit register inside a microcontroller that works independently of the program. On every fourth clock of the oscillator it increments its value until it reaches the maximum (255), and then it starts counting over again from zero. As we know the exact timing between each two increments of the timer contents, timer can be used for measuring time which is very useful with some devices.

CENTRAL PROCESSING UNIT has a role of connective element between other blocks in the microcontroller. It coordinates the work of other blocks and executes the user program.

2.1.2 Application of PIC16F84

PIC16F84 perfectly fits many uses, from automotive industries and controlling home appliances to industrial instruments, remote sensors, electrical door locks and safety devices. It is also ideal for smart cards as well as for battery supplied devices because of its low consumption.

EEPROM memory makes it easier to apply microcontrollers to devices where permanent storage of various parameters is needed (codes for transmitters, motor speed, receiver frequencies, etc.). Low cost, low consumption, easy handling and flexibility make PIC16F84 applicable even in areas where microcontrollers had not previously been considered (example: timer functions, interface replacement in larger systems, coprocessor applications, etc.).

In System Programmability of this chip (along with using only two pins in data transfer) makes possible the flexibility of a product, after assembling and testing have been completed. This capability can be used to create assembly-line production, to store calibration data available only after final testing, or it can be used to improve programs on finished products.

2.2 HARDWARE OF PIC16F84

There are also 13 I/O pins that are user-configured on a pin-to-pin basis. Some pins are multiplexed with other device functions. These functions include:

• External interrupt

• Change on PORTB interrupt

• Timer0 clock input

Figure 2.3 shows pin diagram of the PIC16F84A and Table 2.1 details the description of each pin.

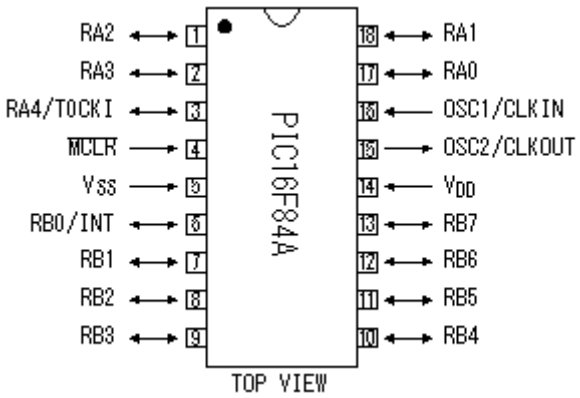


Figure 2.3 : PIC16F84A Pin Diagram

OSC1/CLKIN	:	Oscillator crystal input. External clock source input.
OSC2/CLKOUT	:	Oscillator crystal output. Connects to crystal or resonator in crystal oscillator mode.
MCLR(inv)	:	Master clear(reset)input. Programming voltage input. This pin is an active low reset to the device.
RA0 - RA3	:	Bi-directional I/O port.
RA4/T0CKI	:	Bi-directional I/O port. Clock input to the TMR0 timer/counter.
RB0/INT	:	Bi-directional I/O port. External interrupt pin.
RB1 - RB7	:	Bi-directional I/O port.
V_{SS}	:	Ground

V_{DD}	:	Positive supply(+2.0V to +5.5V)

Table 2.1 : PIC16F84A pin explanation

2.2.1 PIC16F84A Block Diagram

The PIC16F84A belongs to the mid-range family of the PICmicro® microcontroller devices. A block diagram of the device is shown in Figure 2.4. The program memory contains 1K words, which translates to 1024 instructions, since each 14-bit program memory word is the same width as each device instruction. The data memory (RAM) contains 68 bytes. Data EEPROM is 64 bytes. Figure 2.4 shows the block diagram for PIC16F84A. (John Iovine,2004)

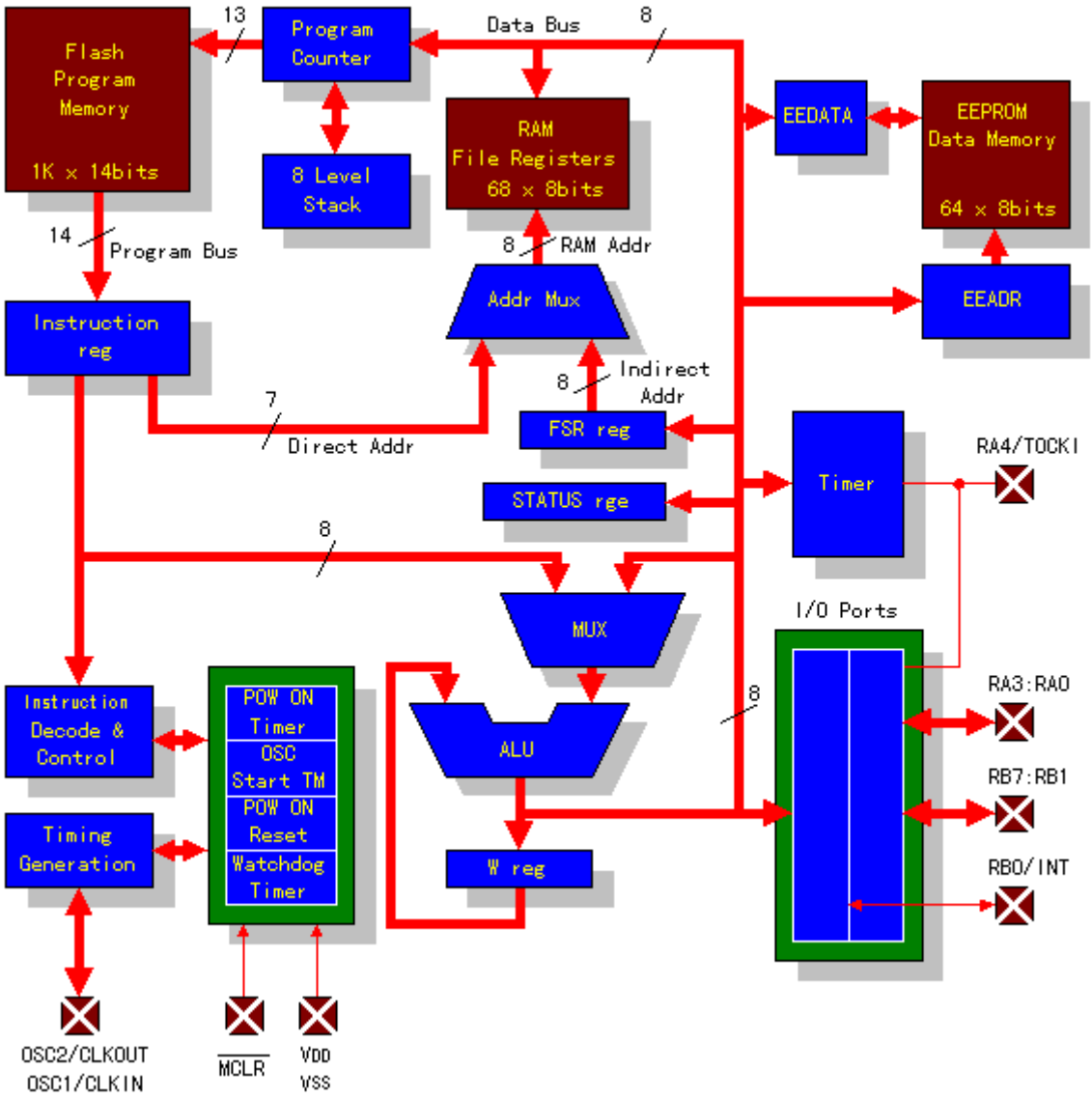


Figure 2.4 : PIC16F84A block diagram

(a) Flash Program Memory

Flash memory is used to store the program. One word is 14 bits long and 1024 words (1k words) can be stored. Even if power is switched off the contents of the flash memory will not be lost. Flash memory can be written to using the writer, but the number of times it be rewritten is limited to 1000 times. The flash memory block diagram is shown in Figure 2.5. (John Iovine 2004)

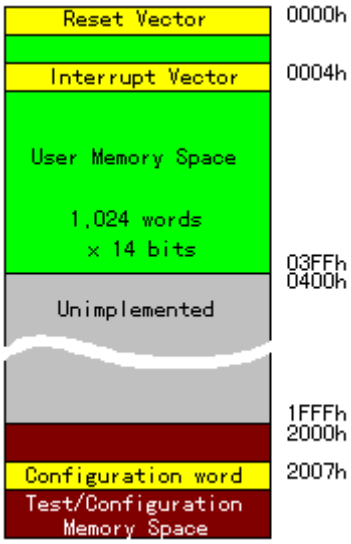


Figure 2.5 : Flash memory block diagram

The usage of some program memory addresses is already decided.

- **Reset Vector (0000h)**

  When a reset is executed, either by turning power on, by the WDT (Watchdog Timer) or any other factor, the program will start from this address.

- **Peripheral Interrupt Vector (0004h)**

  When there is a time-out interruption from the timer (TMR0) or an outside interrupt, the program will start from this address.

- **Configuration word (2007h)**

The basic operation of the PIC is specified at this memory location. The enable bits of the Power-up timer, and the Watch-dog timer as well as the oscillator selection bits are set here.

(b) RAM(Random Access Memory) File Registers

- Bank switching is used to access this memory. Each bank has a memory capacity of 80 bytes (00h-4Fh). The PIC16F84A has two banks.
- This memory is divided into two sections.
- The first 12 bytes (00h-0Bh) of each bank are called SFR (Special Function Registers) and are used to record the operating states of the PIC, the input/output (I/O) port conditions and other conditions.
- There are 16 different registers in the SFR (11 in bank 0 and 5 in bank1). The content of each register is managed by the PIC. Although there are a total of 24 file registers, several of them are in both banks.
- The remaining 68 bytes (0Ch-4Fh), from byte 13 upward, are called GPR (General Purpose Registers) and can be used to temporally store results and conditions while the program is running.
- The contents of the GPR are the same in both banks, so even with bank switching the total capacity is only 68 bytes.
- The contents of the GPR are lost when the power is switched off. There is no limit to the number of times data can be rewritten.

(c) EEPROM(Electrically Erasable Programmable Read Only Memory)

This is nonvolatile memory, the contents is not lost when power is turned off. The memory contents can be rewritten by the program. The total capacity is 64 bytes, and the number of times it can be rewritten to is limited to about one million. For this reason it is not recommended for temporary data storage. It is used to store data which will not change frequently. Data can be safely stored for 40 years.

(d) SFR Registers

16 different SFR (Special Function Registers) can be specified by the bank switching technique. Figure 2.6 shows the RAM File Registers. The memory capacity is only 160 bytes. The contents of the registers with the left pointing arrow are the same on both banks. The other registers of the SFR are accessible through bank switching.. Table 2.2 shows the function of each SFR.

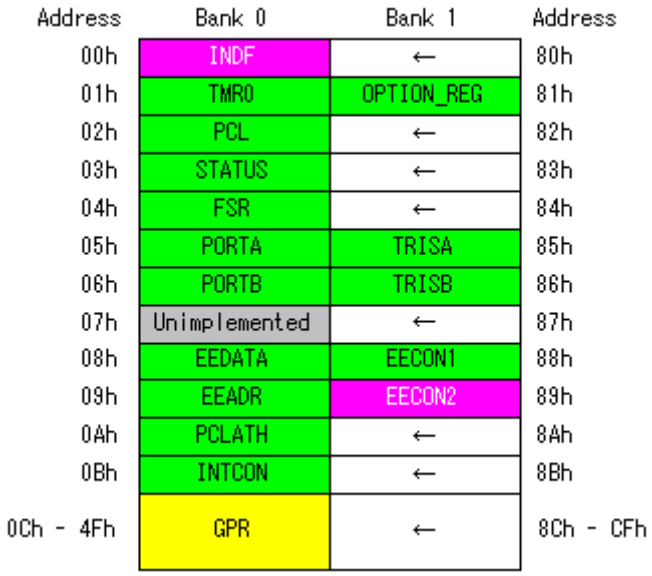


Figure 2.6 : SFR register block diagram

Each SFR has the following function.		
INDF	:	Data memory contents by indirect addressing
TMR0	:	Timer counter
PCL	:	Low order 8 bits of program counter
STATUS	:	Flag of calculation result
FSR	:	Indirect data memory address pointer
PORTA	:	PORTA DATA I/O
PORTB	:	PORTB DATA I/O
EEDATA	:	Data for EEPROM
EEADR	:	Address for EEPROM
PCLATH	:	Write buffer for upper 5 bits of the program counter
INTCON	:	Interruption control
OPTIN_REG	:	Mode set
TRISA	:	Mode set for PORTA
TRISB	:	Mode set for PORTB
EECON1	:	Control Register for EEPROM
EECON2	:	Write protection Register for EEPROM

Table 2.2 : SFR register funtions

(e) Program Counter

This counter contains the address of the next instruction (fetch address) to be read from the program memory (flash memory). It is a 13 bit counter. Generally the count increments each time an instruction is executed, and the location of the following instruction is shown in the PC. But when a jump is executed the contents of this counter is rewritten to that of the jump address.

(f) 8 Level Stack

The stack stores the program return address when a jump is executed.

For example if the same instructions are to be executed more than once, a subroutine is used. A RETURN instruction signifies the end of the subroutine and the program continues were it left off. A CALL instruction tells the program to jump to a subroutine. When the CALL instruction is encountered, the return address (the address of the instruction immediately after the CALL) is stored at the top of the stack. This operation is sometimes called a PUSH. When the subroutine processing is finished and the RETURN instruction is executed, the address at the top of the stack is put in the PC and the program continues normal execution. This operation is sometimes called a POP. This way multiple subroutines can be called and the processing will return to the part of the program which called the subroutine. Since there are eight stack registers, eight subroutine calls can be made sequentially. After the eighth, a call will roll the contents of the eighth (and last register) back to the top of the stack. When a return is executed, the PC will send the program to the wrong address, and the program will not work properly. For this reason only eight sequential subroutines calls can be made.

A return from a subroutine call must be done by the RETURN instruction. Never use the JUMP instruction to return from a subroutine.

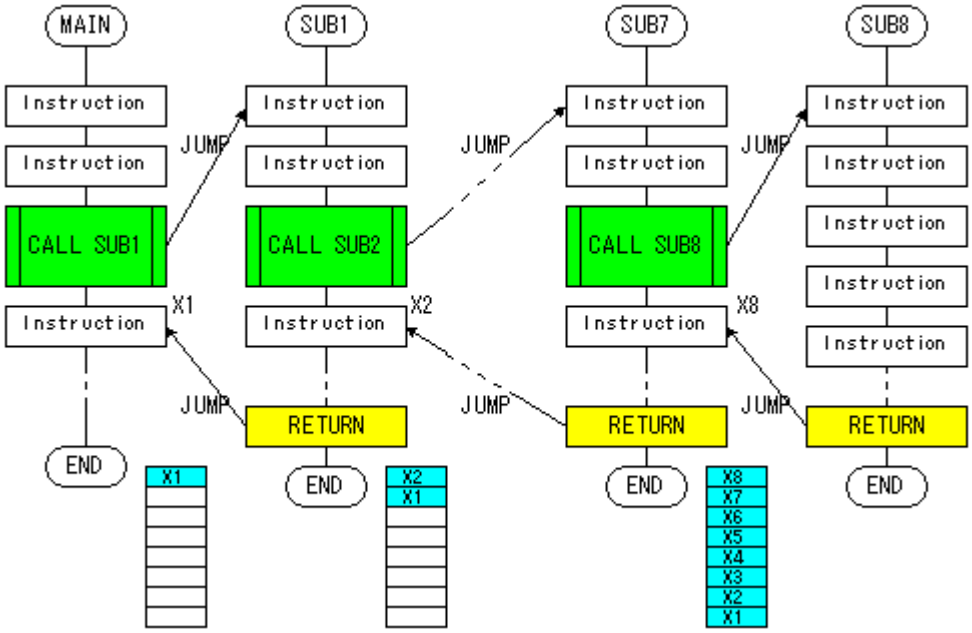


Figure 2.7 : 8 level stack block diagram

(g) Instruction Register

The instruction at the address specified by the PC is read to this register. This operation is called a FETCH.

(h) Instruction Decode & Control

The instruction in the Instruction Register is analyzed here, and (according to its contents) the operation is performed.

(i) Multiplexer and Arithmetic Logic Unit

The calculations are performed by the Multiplexer and Arithmetic Logic Unit (ALU).Without these two it would not be a computer.

(j) W Register

This is the working register. It is used to temporarily store the contents of the ALU. It is indispensable for calculation operations. The contents of this register can be transferred to the various registers to be utilized by the program. It is also used to control the I/O ports.

(k) STATUS Register

This Register stores the result of the ALU (Zero, positive or negative), a time-out condition, register bank select etc.

(l) FSR Register

FSR (File Select Register) is used to specify the address of the RAM file register when using the indirect addressing method.

In the direct address method the register address is specified by the program instructions. In this case a 7 bit address, from 0 to 127, can be specified. This range is for one bank. To change the bank, the RP0 bit of the status register must also be specified. Since the FSR is 8 bits, it is possible to specify the address and bank at the same time, saving instruction cycles. In the PIC16F84A memory locations 80 (50h) to 127 (7Fh) are not installed.

When using the FSR it is convenient to make the file register data area continuous. Incrementing the FSR simplifies processing during read and write operations.

(m) Address Multiplexer

Distinguishes between direct and indirect addressing.

(n) EEDATA

This register is used when reading from, or writing data to the EEPROM.

(o) EEADR

This register specifies the EEPROM address. Since it is composed of eight bits addresses from 0 to 255 can be specified. The PIC16F84A has only 64 bytes of memory installed. The address of EEPROM is started from 2100h.

When writing data during program execution it is necessary to sequentially write 55h then AAh to the EECON2 register.

(p) Timer

The PIC16F84A has only one timer (TMR0, an 8 bit timer). It times out when the count reaches 256 and the TOIF bit of the INCON register of the SFR becomes "1". With a timed out condition it is possible to make an interrupt occur. The interrupt will stop the processing which was in progress at that time. To make the interrupt occur the GIF and the TOIE bits of the INTCON register of the SFR must be set (1)

(q) I/O Ports

There are 13 I/O pins, with individual direction controls. The mode (input or output) of each pin can be set from within the program. The 13 pins are divided into two groups. The A Port has five pins and the B Port has eight. There is a limitation on control timing, but each of the 13 pins can be individually controlled.

(r) Timing Generation

This circuit generates the clock pulses which determine the speed of operation. Oscillator operation is determined by external capacitors and a crystal (or ceramic) oscillator. If high stability is desired a crystal or ceramic resonator is used. Generally the circuitry is simpler with the resonator, which incorporates the ceramic and capacitors in one module. An external clock can also be used.

The PIC16F84A executes one instruction (one cycle) every four clock pulses, using

pipeline architecture. But when a JUMP is executed two cycles are necessary (8 clock pulses). With a 20MHz clock the execution time of an instruction is 200ns. (The pulse period of 20MHz is 50ns (1 / 20MHz = 50ns), so 50ns x 4 cycles = 200ns). 5,000,000 instructions can be executed in one second. (2,000,000 / 4 = 5,000,000)

(s) Initialization circuits

- The PIC16F84A has various initialization circuits.
- POW ON Timer : When power is turned on this timer inhibits operation until the voltage is stable.
- OSC StartTimer : When power is turned on this timer inhibits operation until the clock is stable.
- POW ON Reset : When power is turned on this timer initializes the inner circuitry of the PIC.
- Watchdog Timer : This timer watches over the normal operation of the PIC software. It must be regularly cleared by software. When it times out the PIC returns to the point in the program immediately after power on. This timer is used to recover program operation when the software has a defect (or bug). Even if the program is reinitialized the bug is still in the program.

2.3 SOFTWARE OF PIC16F84A

PIC can do various operation by executing the instructions which were stored in the program memory(PM). The instruction which is stored at the head(0000h) of PM is executed when turning on the power. If GOTO instruction is stored there, the PIC changes the execution address of the program according to the contents of the instruction. To make software is to make the order of the operation to make PIC do.

2.3.1 General format for instructions

The format for instructions of PIC16 series is the following three kinds as shown in Figure 2.8. The instructions are written in the program memory and one instruction is composed of 14 bits. These 14 bits are called a word.

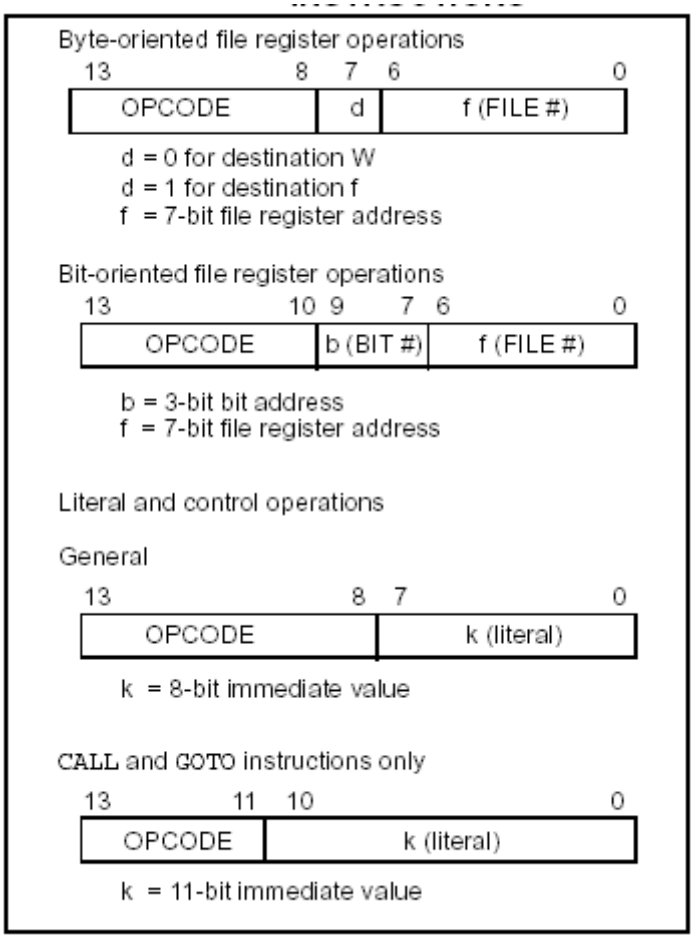


Figure 2.8 : General format for instruction

For byte-oriented instructions, ’f’ represents a file register designator and ’d’ represents a destination designator. The file register designator specifies which file register is to be used by the instruction.The destination designator specifies where the result of the operation is to be placed. If ’d’ is zero, the result is placed in the W register. If ’d’ is one, the result is placed in the file register 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 of the file in which the bit is located.
- For literal and control operations, ’k’ represents an eight or eleven bit constant or literal value.

2.3.2 Instruction set of PIC16F84

Complete set which includes 35 instructions is given in Table 2.3. A reason for such a small number of instructions lies primarily in the fact that we are talking about a RISC microcontroller whose instructions are well optimized considering the speed of work, architectural simplicity and code compactness. The only drawback is that programmer is expected to master "uncomfortable" technique of using a reduced set of 35 instructions.

All instructions are executed in one cycle except for conditional branch instructions if condition was true, or if the contents of program counter was changed by some instruction. In that case, execution requires two instruction cycles, and the second cycle is executed as NOP (No Operation). Four oscillator clocks make up one instruction cycle. If we are using an oscillator with 4MHz frequency, the normal time for executing an instruction is 1 μs, and in case of conditional branching, execution period is 2 μs. (Microchip Technology, 2001)

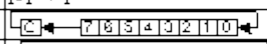
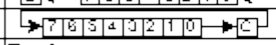
Mnemonic		Description	Operation	Flag	Cycle	Notes
		Data transfer				
MOVLW	k	Move constant to W	k → W		1	
MOVWF	f	Move W to f	W → f		1	
MOVF	f, d	Move f	f → d	Z	1	1,2
CLRW	-	Clear W	0 → W	Z	1	
CLRF	f	Clear f	0 → f	Z	1	2
SWAPF	f, d	Swap nibbles in f	f(7:4), (3:0) → f(3:0),(7:4)		1	1,2
		Arritmetic and logic				
ADDLW	k	Add constant and W	W+1 → W	C,DC,Z	1	
ADDWF	f, d	Add W and f	W+f → d	C,DC,Z	1	1,2
SUBLW	k	Subtract W from constant	W-k → W	C,DC,Z	1	
SUBWF	f, d	Subtract W from f	W-f → d	C,DC,Z	1	1,2
ANDLW	k	AND constant with W	W.AND.k → W	Z	1	
ANDWF	f, d	AND W with f	W.AND.f → d	Z	1	1,2
IORLW	k	OR constant with W	W.OR.k → W	Z	1	
IORWF	f, d	OR W with f	W.OR.f → d	Z	1	1,2
XORLW	k	Exclusive OR constant with W	W.XOR.k → W	Z	1	1,2
XORWF	f, d	Exclusive OR W with f	W.XOR.f → d	Z	1	
INCF	f, d	Increment f	f+1 → f	Z	1	1,2
DECF	f, d	Decrement f	f-1 → f	Z	1	1,2
RLF	f, d	Rotate Left f trough carry		C	1	1,2
RRF	f, d	Rotate Right f trough carry		C	1	1,2
COMF	f, d	Complement f	F → d	Z	1	1,2
		Bit operations				
BCF	f, b	Bit Clear f	0 → f(b)		1	1,2
BSF	f, b	Bit Set f	1 → f(b)		1	1,2
		Directing a program flow				
BTFSC	f, b	Bit Test f, Skip if Clear	jump if f(b)=0		1 (2)	3
BTFSS	f, b	Bit Test f, Skip if Set	jump if f(b)=1		1 (2)	3
DECFSZ	f, d	Decrement f, Skip if 0	f-1 → d, jump if Z=1		1(2)	1,2,3
INCFSZ	f, d	Increment f, Skip if 0	f+1 → d, jump ifZ=0		1(2)	1,2,3
GOTO	k	Go to address	W.AND.k → W		2	
CALL	k	Call subroutine	W.AND.f → d		2	
RETURN	-	Return from Subroutine	W.OR.k → W		2	
RETLW	k	Return with constant in W	W.OR.f → d		2	
RETFIE	-	Return from interrupt	W.XOR.k → W		2	
		Other instructions				
NOP	-	No Operation			1	
CLRWDT	-	Clear Watchdog Timer	0 → WDT,1→TO,1 → PD	$\overline{TO}$, $\overline{PD}$	1	
SLEEP	-	Go into standby mode	0 → WDT,1→TO,0 → PD	$\overline{TO}$, $\overline{PD}$	1	

Table 2.3 : Instruction Set of PIC16F84

CHAPTER 3 : HARDWARE OF PROJECT

3.1 Circuit Explanation of Digital Alarm Clock

In this part, how the circuit work and also the function of every parts of the circuit will be explain. The circuit is shown in Figure 3.1. Basically, the circuit can divide to 5 parts with different function.

i. Microcontroller PIC

ii. Power Supply

iii. Time display

iv. Alarm circuit

v. Input signal circuit

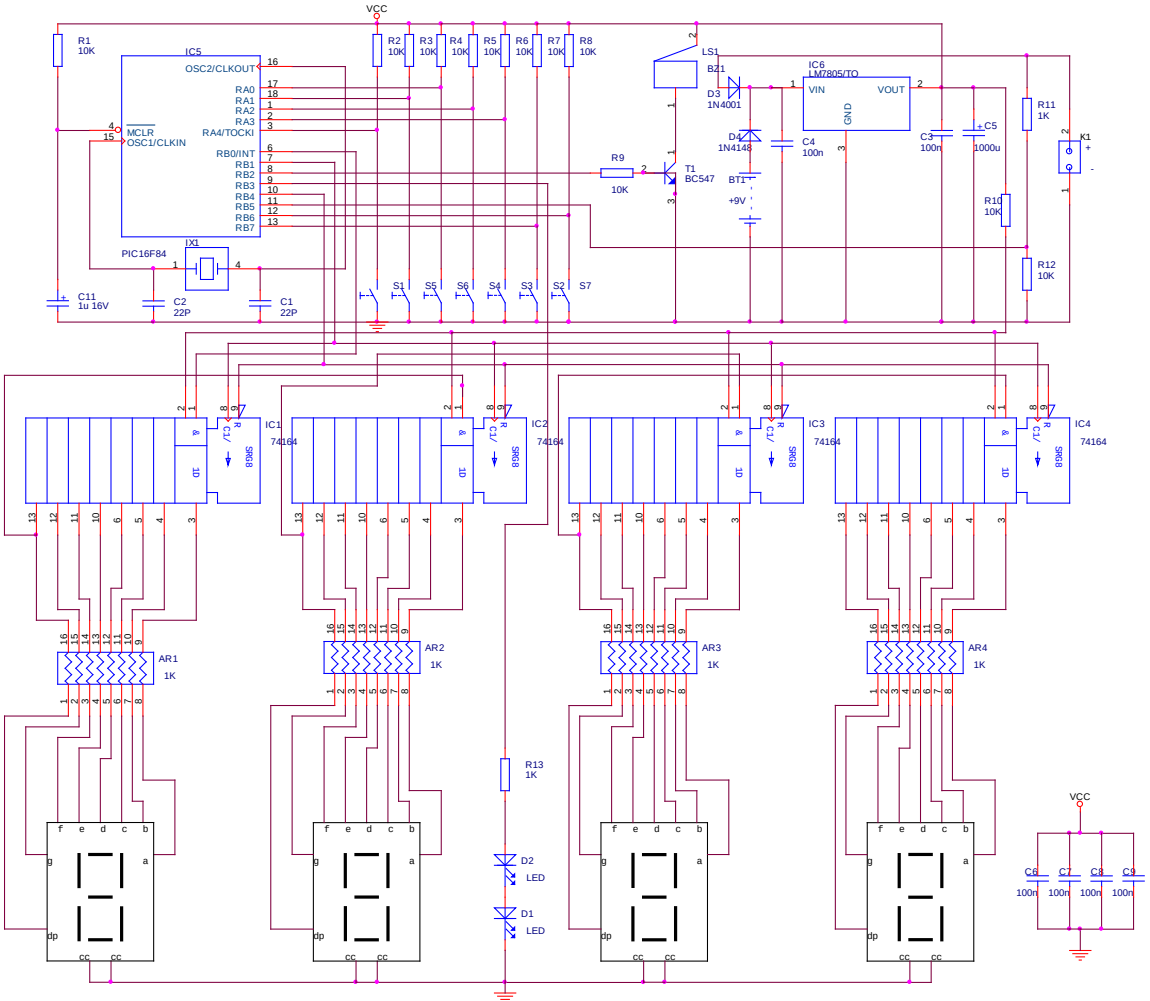


Figure 3.1 : Schematic of digital alarm clock

3.2 Power Supply

Circuit diagram above shows the power supply for digital alarm clock. The 7805 regulator is used to get a stable, accurate, known voltage (+5V) for the circuits.

In this power supply circuit, the input to the 7805 regulator is around 12VDC, and the regulation voltage is 5VDC, so there is plenty of headroom.

Capacitor C5 (1000uF) is used to smooth the output voltage of the regulator to make sure the output voltage as clean as possible without ripple.

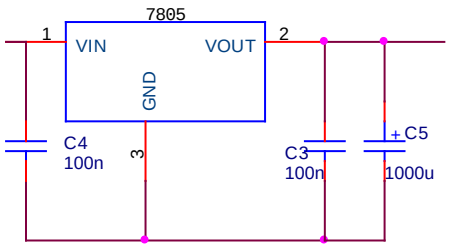


Figure 3.2 (a) : Schematic of power supply

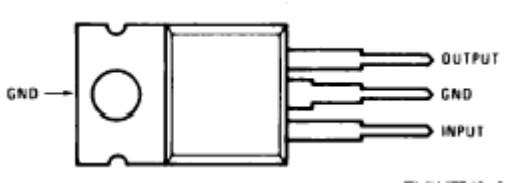


Figure 3.2(b) : Regulator 7805

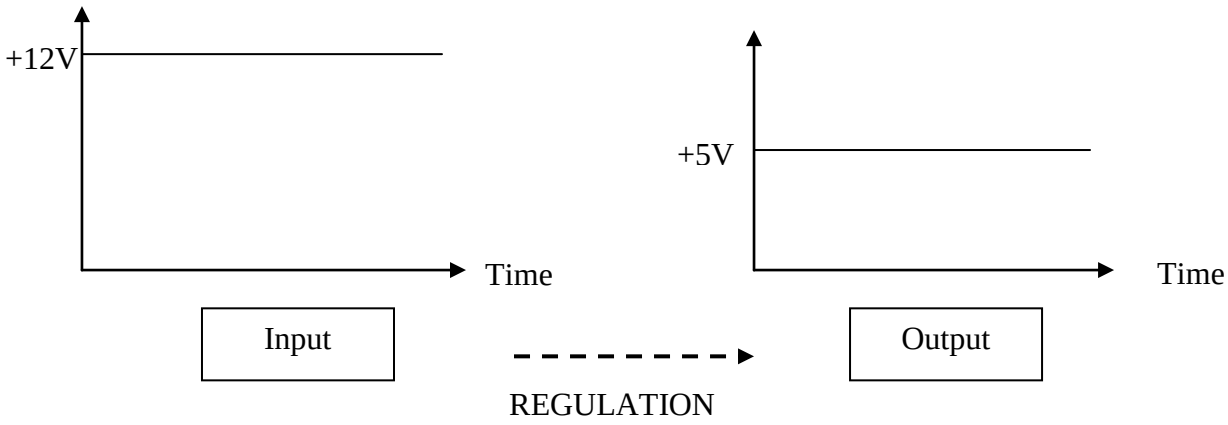


Figure 3.3 : Input and output of power supply.

In use, a regulator IC will get quite hot, so a heatsink will need to be attached to it to dissipate the heat. Heatsinks are rated by their 'thermal resistance' (R_{th}) given in °C/W. For example, a rating of 2°C/W means the heatsink (and therefore the regulator attached to it) will be 2°C hotter than the surrounding air for every 1W of heat it is dissipating. Lower thermal resistance means a better heatsink. The type of heatsink was chose for this circuit is shown in Figure 3.4:

Figure 3.4 : Heatsink

3.3 Microcontroller PIC

The PIC microcontroller (shown in Figure 3.5) is the most important part of this circuit. All intelligence logic is vested in the PIC16F84A MCU in position IC5. This MCU have two bi-directional I/O ports which is PORTA and PORTB. In this circuit, RA0-RA4 and RB5-RB7 are used as input port lines and RB0-RB4 as output port lines, a fair amount of executable code run from the on-chip memory is able to take total control of the circuit, requiring just a 5V supply voltage and a clock signal generated with the aid of an external 4-MHz quartz crystal, X1 as shown in Figure 3.6

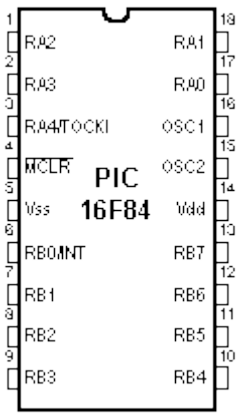


Figure 3.5 : PIC microcontroller pin diagram

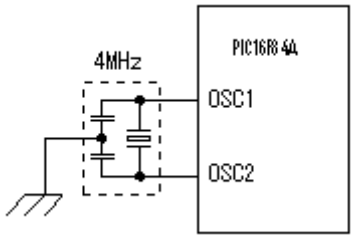


Figure 3.6 : External 4MHz quartz crystal

The most important job of the microcontroller is to produce an accurate time to this circuit. It is based on the microcontroller TIMER_0 which has ability to interrupt the main program and deviate its execution to an interrupt handing routine whenever its count overflows (FFh-00h)

This microcontroller is configured so that the TIMER_0 overflow will take place every 2.048 ms. This is accomplished by setting the prescaler to divide the TIMER_0 clock by eight.

Whenever called, the interrupt handling routine, will increment a 16-bit counter. This counter has a maximum count of 29.297 (7271h). This means that every minute (29,297 X 2.048ms = 60,000.256ms or close to one minite) it will overflow.

3.4 Reset Pin of Microcontroller

A Power-on Reset pulse is generated on-chip when VDD rise is detected (in the range of 1.2V - 1.7V). To take advantage of the POR (Power On Reset), just tie the MCLR pin directly (or through a resistor) to VDD. This will eliminate external RC components usually needed to create Power-on Reset. A minimum rise time for VDD must be met for this to operate properly.

When the device starts normal operation (exits the RESET condition), device operating parameters must be met to ensure operation. If these conditions are not met, the device must be held in RESET until the operating conditions are met.

In the circuit as shown in Figure 3.7, the microcontroller needs to reset every time the power on. In this circuit, component R1 and C11 are used to guarantee that the microcontroller is reset at power-on.

R1 (10K) is to make sure that voltage drop across R does not exceed 0.2V (max leakage current spec on MCLR pin is 5 mA). A larger voltage drop will degrade VIH level on the MCLR pin.

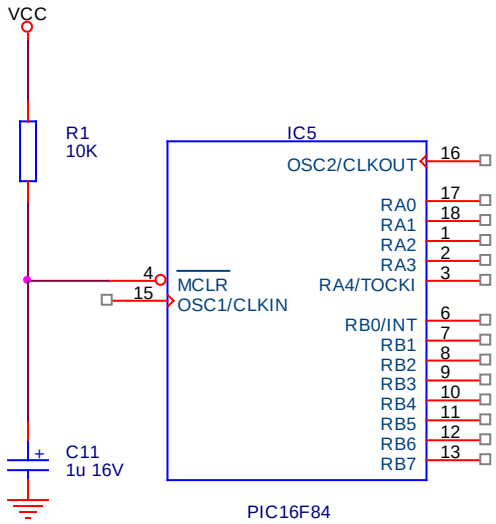


Figure 3.7 : Reset pin microcontroller

3.5 Input Port Of Microcontroller

The microcontroller is set to be active LOW, so a pull-up resistor (10k) is connected to the input as shown in Figure 3.8 to ensure that the High logic level at all inputs when the associated switches is open as shown in Figure 3.9.

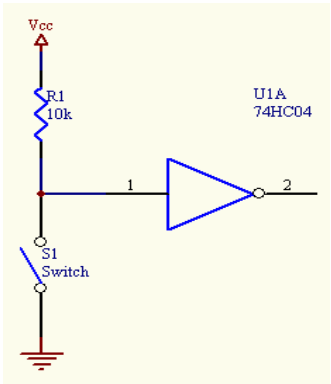


Figure 3.8 : Pull-up resistor

A pull-up resistor is connected to the input as shown in Figure 3.8. This resistor is to limit the amount of current that can flow through the circuit.

When switch S1 is opened (off), pin 1 is tied to Vcc through the resistor. Since pin1 is a high impedance input, placing a voltmeter or logic probe on pin 1 will show Vcc (+5v) presents at pin 1.

When switch S1 is closed (on), pin 1 has a direct connection to GND, which takes it to the low state. The pin1 side of R1 also has a direct connection to ground. Current will flow from Vcc, through R1, and to ground. It isn't considered a short, however, because R1 will limit the amount of current that can flow to a very small amount. In fact, it can be computed using Ohms law.

$$I = V / R$$

$$I = 5v / 10{,}000ohms$$

$$I = .0005A\ (.5mA)$$

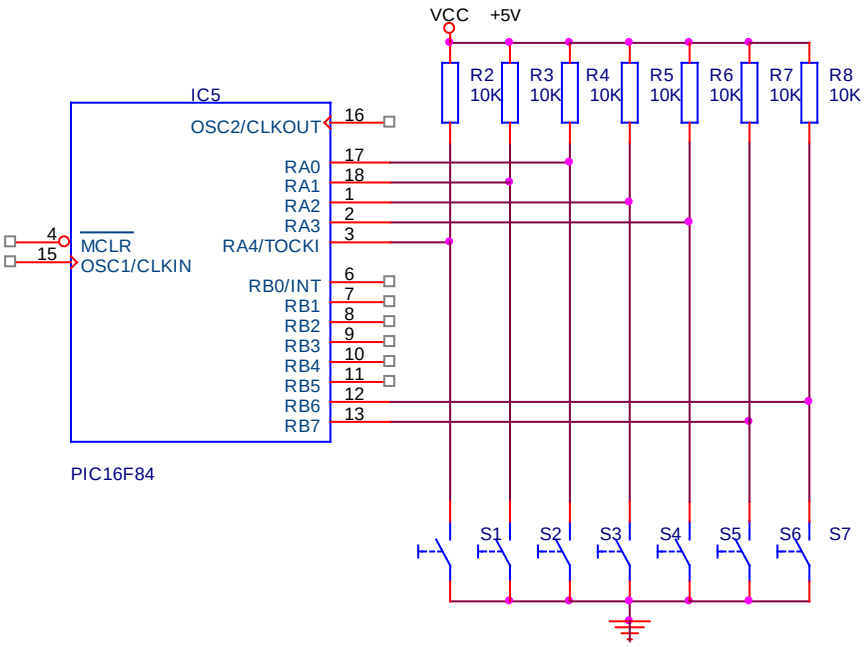


Figure 3.9 : Input port of microcontroller

Switches S1-S5 are connected to the microcontroller's PORTA pins (all configured as inputs). Resistors R2-R6 guarantee a High logic level at PORTA pins when the switches are open. When closed, these keys will force a Low state on the PORTA pin they connected to, triggering the execution of specific clock control routines.

PORTB.6 and PORTB.7 are also configured as inputs. R7 and R8 guarantee the High logic level at these inputs when the associated switch S6 or S7 is open. S6, when closed, will active the 'Snooze' function. Similarly, S6 will temporarily active the display. This function is only available when the clock is operating in the battery mode.

3.6 Voltage Divider

Resistor R11 and R12 form a voltage divider fed by the main 12 VDC source as shown in Figure 3.10. They guarantee a High logic level at PORTB.5 when the main supply voltage is available. In this situation the display will be permanently on. If for some reason the main power is not available (AC power failure) there will be a Low logic level at PORTB.5 and the entire display will be turned off by software.

Voltage at PORTB.5 when the main power is available:

$$V_{b5} = \frac{R12}{R12 + R11} \times V_{in}$$

$$V_{b5} = \frac{2.2k}{2.2k + 1k} \times 12V = 8.25V$$

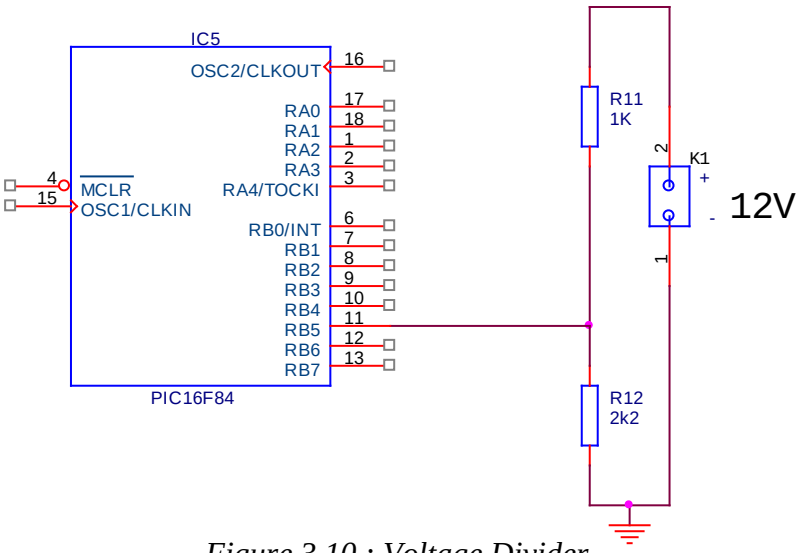


Figure 3.10 : Voltage Divider