Pic Microcontroller

Sabri KOCER
Necmettin Erbakan University

Resul BUTUNER
Konya Adil Karaagac Vocational and Technical Anatolian High School

Introduction

In simple applications, non-programmable solutions that do not require software development are preferable. Writing software is time consuming so it is more expensive and in simple and / or short run applications it is often more reasonable to perform tasks in hardware. However, as the complexity of the system increases, the benefits of using programmable systems increase. One of the main advantages of programmable systems is their flexibility, allowing you to update the operation of a system just by changing the program without having to redesign the hardware. This flexibility is very important, allowing products to be updated easily and economically.

The impressive effort of Microchip dedicated to updating its products has led to a massive commercialization of new models that have raised it to first place in the ranking 8-bit microcontroller world since 2002.

It is now possible to program microchips. Thanks to this technology, complex circuits can be made smaller by the use of these micro-controllers, of which pic is an excellent example. However, cheaper and smaller circuits would be possible using the PIC microcontroller. However, even complex logic circuits can benefit greatly from the use of PIC microcontrollers. In addition, by making changes to a PIC program, it will be easier to create prototypes without changing circuit designs and electronic components.

What has made the PIC MCU successful is:

- The availability of excellent, low-cost (free) development tools;
- The largest and strongest user Internet based community of probably any silicon chip family;
- An outstanding distributor network with a wide variety of parts available in very short notice;
- A wide range of devices with various features that just about guarantees that there is a PIC microcontroller suitable for any application;
- Microchip's efforts at continually improving and enhancing the PIC MCU family

based on customer's needs.

This new updated and expanded edition includes, in addition to the PIC16F87X mid-range microcontrollers, the improved PIC18F range that are having a great acceptance and contains an introduction to modern 16-bit microcontrollers of the PIC24F and PIC24H families, as well as Digital Signal Controllers (DSC) embodied in the dsPIC families.

Importance of PIC Microcontrollers

Microchip, manufacturer of PIC microcontrollers, has remained a leader since 2002 8-bit microcontroller sales worldwide. MCU sales and units shipments driven by the spread of embedded control in systems, more sensors, and the rush to connect end-use applications to the Internet of Things (IoT).

PIC is a family of microcontrollers manufactured by Microchip Technology, derived from the PIC1650 originally developed by the Microelectronics Division of General Instrument. The name PIC initially referred to Peripheral Interface Controller, and is currently being expanded as Programmable Intelligent Computer. The first parts of the family were available in 1976; By 2013, the company had shipped more than twelve billion individual parts, used in a wide variety of embedded systems.

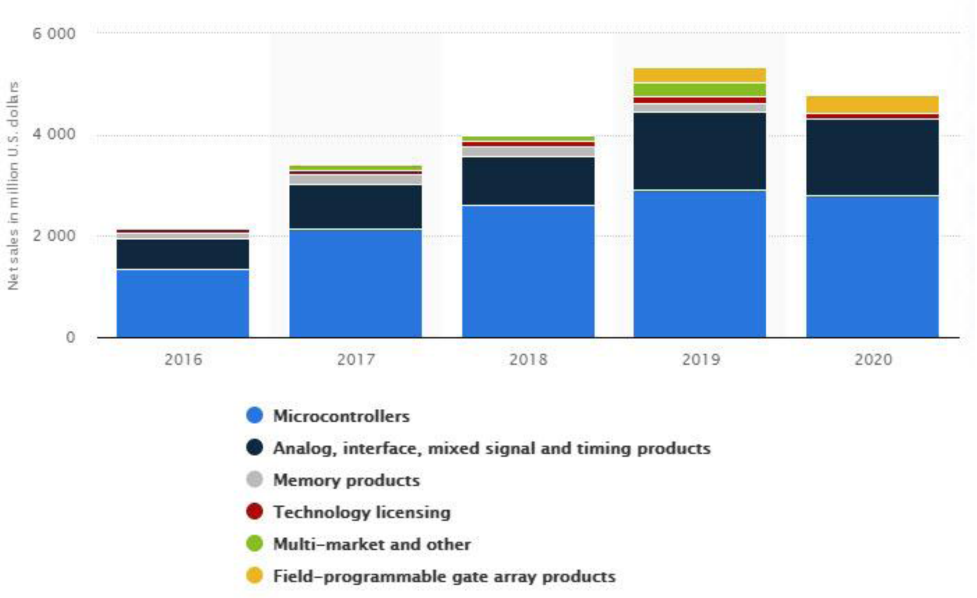


Figure 1. Microchip Technology Net Sales 2016-2020, Based on Product Line Published by Thomas Alsop, July 1, 2020

Figure 1 shows Microchip Technology's net sales from 2016 to 2020, separated by product line. In 2020, Microchip Technology's net microcontroller sales totaled approximately US $ 2.82 billion. Sales of Microchip products are absorbed by 43% by Asia and Japan, while that Europe supports 28% and America 29%. Regarding the market segments to which the PIC microcontrollers are destined, it stands out the generic area of Consumer Electronics with 35%, followed by the automotive industry with 18%.

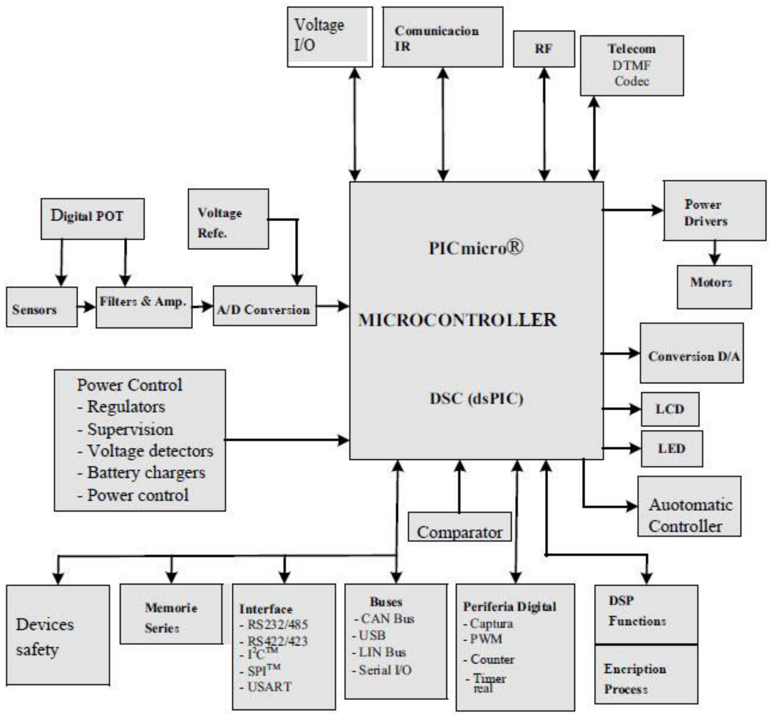


Figure 2. PIC Microcontroller Application Area

The great diversity of microcontroller models allows the designer to find the one that contains all the resources and memory capacities that he needs for his application (Figure 2.).

The technology used in the manufacture of PIC devices has gone from 0.7 microns in 1998 to 0.22 microns in 2006. This has meant a reduction in the supply voltage, which has gone from being between 2 and 5 , 5 V, to operate with 5V, at a range between 2 and 3.6 V, being the nominal of 3.3 V although the I / O continues to work with 5V.

One of the great advantages of Microchip microcontrollers is their “migrability”, which means the possibility of changing the MCU model and moving to a more powerful one with more memory and peripheral capacity, maintaining the distribution of the pins in the package or Pinout, software and instruction set compatibility, peripheral compatibility, and development tool compatibility.

Classification of PIC Microcontrollers

These microcontrollers or MCUs are characterized by their Harvard architecture with independent program and data memories, which allows simultaneous accessibility and diversity in the length of the positions and the size of both memories.

Microcontrollers are generically classified according to the size of the data handled by the instruction set and there are 4 large groups: 4, 8, 16 and 32 bits. Microchip only manufactures 8-bit and 16-bit microcontrollers and as noted is the world leader in first group sales.

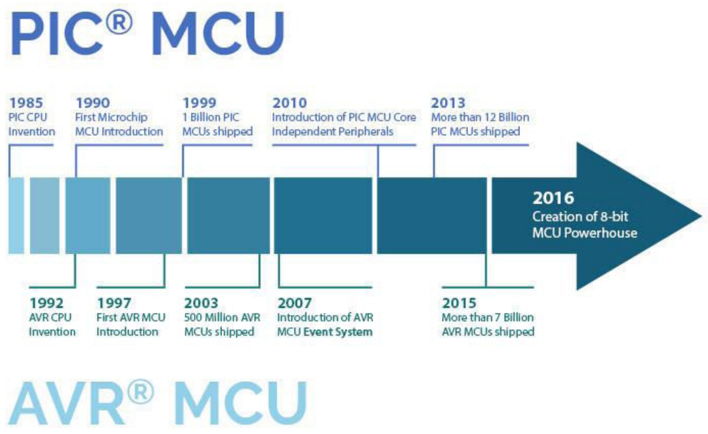

PIC® MCU
1985 PIC CPU Invention
1990 First Microchip MCU Introduction
1999 1 Billion PIC MCUs shipped
2010 Introduction of PIC MCU Core Independent Peripherals
2013 More than 12 Billion PIC MCUs shipped
2016 Creation of 8-bit MCU Powerhouse
1992 AVR CPU Invention
1997 First AVR MCU Introduction
2003 500 Million AVR MCUs shipped
2007 Introduction of AVR MCU Event System
2015 More than 7 Billion AVR MCUs shipped
AVR® MCU

Figure 3. Historical Milestones in 8-bit AVR and PIC MCU Development

The AVR and PIC microcontroller brands, embedded design also represents the most effective architecture. Figure 3 shows historical milestones in 8-bit AVR and PIC MCU development. With 45 years of combined experience in developing commercially available and cost-effective 8-bit MCUs, Microchip is the supplier of choice for many due to its strong legacy and history of innovation in 8-bit.

Most current microprocessors are standard Von Neumann machines. The biggest deviation from this is the Harvard architecture, where instructions and data have separate address, data, and control buses for each memory area, and different memory areas in Figure 4. This directive and data returns can occur at the same time and has the advantages of not having the size of an instruction set relative to the standard data unit (word) size.

The PIC is based on the Harvard architecture in which the program and data buses are kept separate. Early versions of PIC microcontrollers use EEPROM to store program instructions, but have adopted flash memory since 2002 to enable better erasing and storage of code.

Thanks to their simplicity in architecture and ease of use, PIC microcontrollers have proven to be a hit among hobbyists, students and professionals. PIC16F84 and PIC16F877 were some of the most popular PIC microcontrollers with basic functions. Applications that require richer peripherals, higher performance or memory can rely on the PIC18F family.

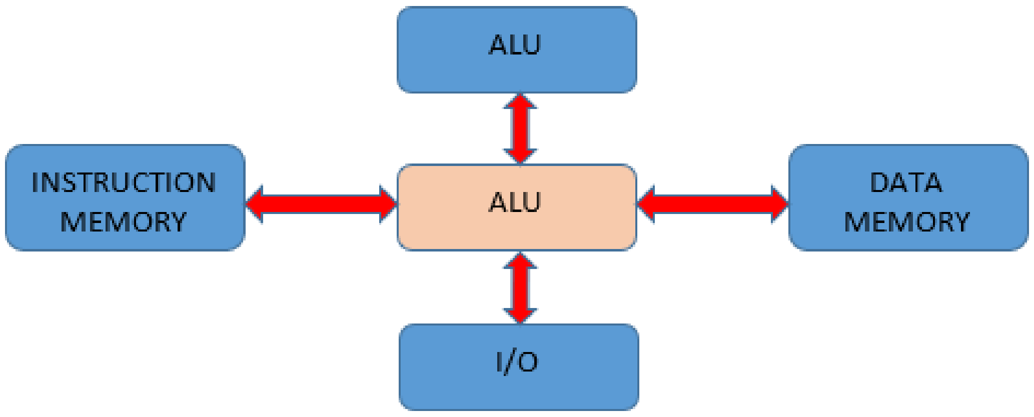


Figure 4. The Harvard Architecture used by PIC Microcontrollers

Instruction bus: Program instructions are fed from the FLASH program memory to the ALU via the 14-bit instruction bus. A 14-bit program word is read to the ALU in each instruction clock cycle

Bus: An 8-bit bus connects the ALU to the data memory area. During each instruction, the ALU can read data from the data memory location, modify the data, then write the data back to the memory.

Instruction Pipeline: The advanced mid-range PIC's dual-bus architecture provides a two-stage instruction pipeline. One each clock cycle execute two instruction phases:

The next instruction is "taken" from Program memory

The current instruction is "executed" and reads/modifies/writes the data memory (if needed)

8-Bit PIC Microcontrollers

They are distinguished by the length of the native data handled by the instructions is 8 bits, which corresponds to the size of the data bus and that of the CPU registers.

8-bit PIC microcontrollers are classified into three broad ranges: Base, Medium and Advanced, with a total of about 300 different models that contain different capacities of memory, peripherals and different types of packages (Figure 5.)

8-bit PIC® microcontrollers come in several core architectures. This can be confusing for someone trying to decide the best option for their product or project. This summary will describe the different options to help you make the right decision.

Microchip offers different ranges of microcontrollers.

- PIC12CXXX base line PIC (8-pin, 12-bit / 14-bit program word):

Low consumption, EEPROM data memory.

- PIC16C5X, low-end or classic (12-bit program word):

Encapsulated 14, 18, 20 and 28 pins, Optimal for applications that work with batteries (low consumption).

- PIC16CXXX, midrange (14-bit program word).

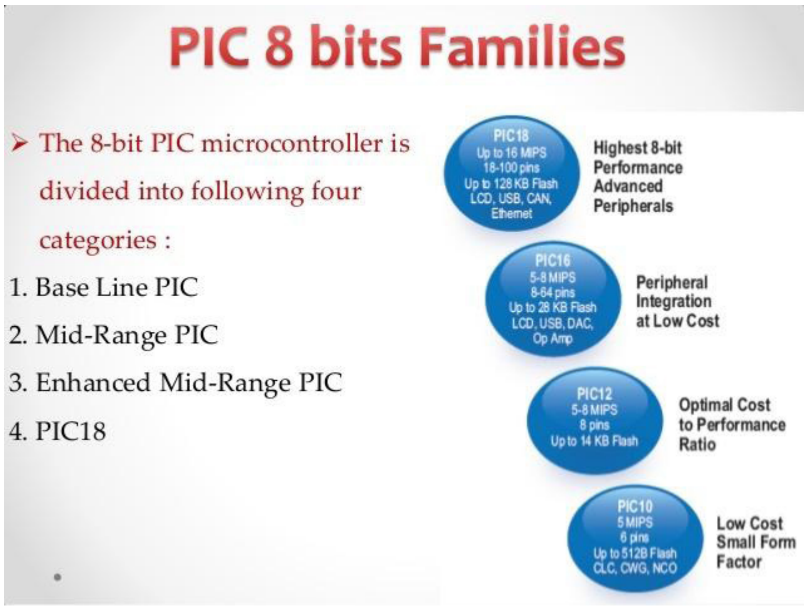
A / D converters and serial port, Encapsulated from 18 to 68 pins.

- PIC17CXXX, Enhanced mid range (16 bit program word).

Open architecture, expandable memory,

- PIC18XXX, high range (16 bit program word).

ny6



PIC 8 bits Families

➢ The 8-bit PIC microcontroller is divided into following four categories :

1. Base Line PIC
2. Mid-Range PIC
3. Enhanced Mid-Range PIC
4. PIC18

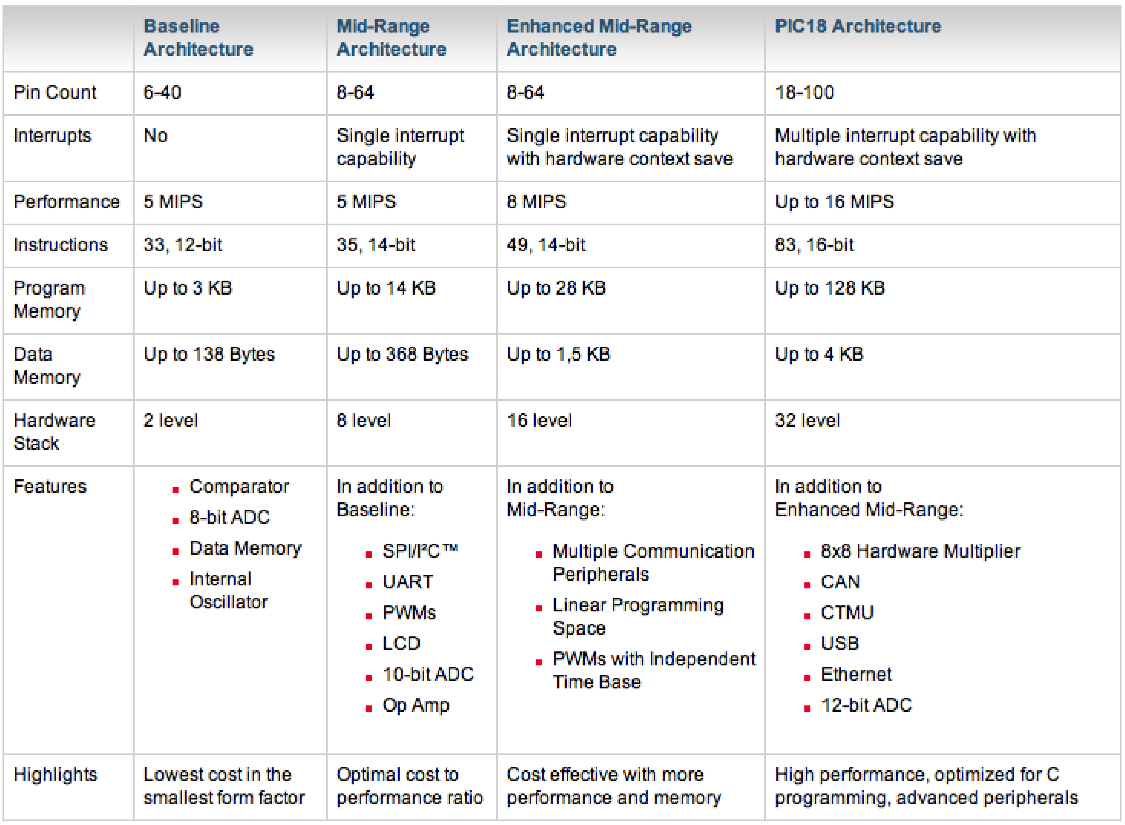
PIC18
Up to 16 MIPS
18-100 pins
Up to 128 KB Flash
LCD, USB, CAN, Ethernet
Highest 8-bit Performance Advanced Peripherals
PIC16
5-8 MIPS
8-64 pins
Up to 28 KB Flash
LCD, USB, DAC, Op Amp
Peripheral Integration at Low Cost
PIC12
5-8 MIPS
8 pins
Up to 14 KB Flash
Optimal Cost to Performance Ratio
PIC10
5 MIPS
6 pins
Up to 512B Flash
CLC, CWG, NCO
Low Cost Small Form Factor

Figure 5. 8-bit PIC Microcontrollers are Classified: Base, Mid-range and Advanced and PIC18

There are currently three main ranges of PIC MCUs: The basic ones (Linebase), the mid-range ones (Mid Range) and the high performance ones (high performance). The PIC18s are considered high performance and have among their members PICs with communication modules and advanced protocols (USB, Ethernet, Zigbee for example).

The dsPICs are Microchip's penultimate release, starting large-scale production in late 2004. They are the first 16-bit inherent data bus PICs. They incorporate all the possibilities of the previous PICs and add several DSP operations implemented in hardware, such as multiplication with accumulator addition (multiply-accumulate, or MAC), barrel shifting, bit reversion or 16x16-bit multiplication.

Microchip Technology launched in November 2007 the new 32-bit microcontrollers with a processing speed of up to 1.6 DMIPS / MHz with USB HOST capability. Its clock frequencies can reach 80MHz from standard 4 to 5MHz quartz thanks to an internal PLL. They operate at 3.3V in their input and output ports, although the manufacturer indicates that except for the pins with analogue function, most voltages of up to 5V are tolerated. They have an optimized architecture with a high degree of parallelism and an M4K-type core and a high capacity of RAM and FLASH ROM memory. All this means that these MCUs allow high information processing.



	Baseline Architecture	Mid-Range Architecture	Enhanced Mid-Range Architecture	PIC18 Architecture
Pin Count	6-40	8-64	8-64	18-100
Interrupts	No	Single interrupt capability	Single interrupt capability with hardware context save	Multiple interrupt capability with hardware context save
Performance	5 MIPS	5 MIPS	8 MIPS	Up to 16 MIPS
Instructions	33, 12-bit	35, 14-bit	49, 14-bit	83, 16-bit
Program Memory	Up to 3 KB	Up to 14 KB	Up to 28 KB	Up to 128 KB
Data Memory	Up to 138 Bytes	Up to 368 Bytes	Up to 1,5 KB	Up to 4 KB
Hardware Stack	2 level	8 level	16 level	32 level
Features	▪ Comparator ▪ 8-bit ADC ▪ Data Memory ▪ Internal Oscillator	In addition to Baseline: ▪ SPI/I²C™ ▪ UART ▪ PWMs ▪ LCD ▪ 10-bit ADC ▪ Op Amp	In addition to Mid-Range: ▪ Multiple Communication Peripherals ▪ Linear Programming Space ▪ PWMs with Independent Time Base	In addition to Enhanced Mid-Range: ▪ 8x8 Hardware Multiplier ▪ CAN ▪ CTMU ▪ USB ▪ Ethernet ▪ 12-bit ADC
Highlights	Lowest cost in the smallest form factor	Optimal cost to performance ratio	Cost effective with more performance and memory	High performance, optimized for C programming, advanced peripherals

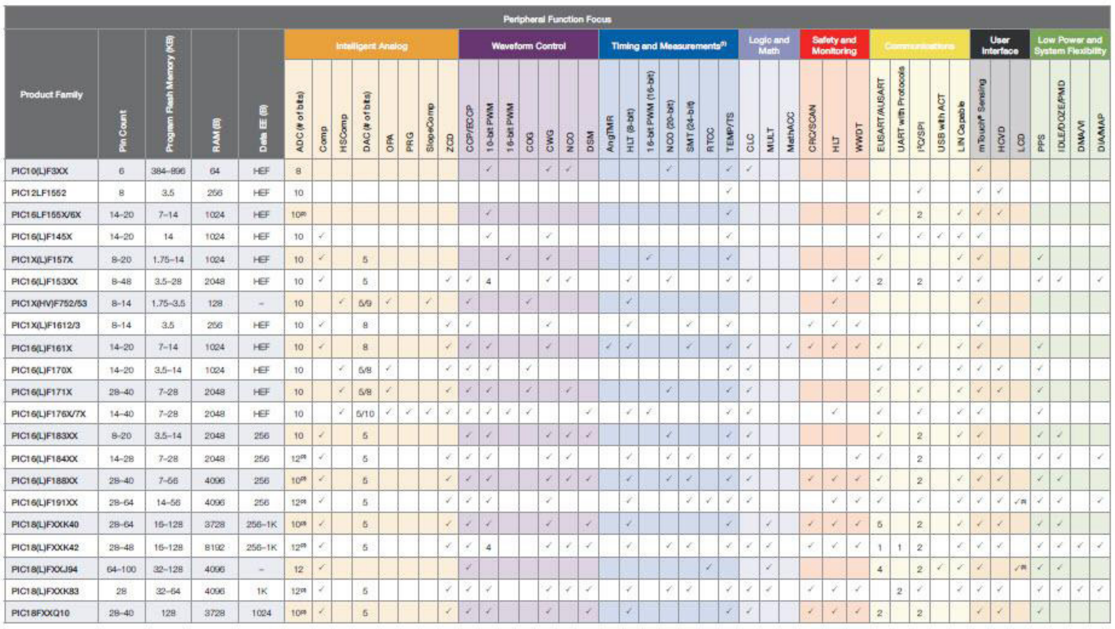
Figure 6. The Differences Among the 8-bit PIC® MCU Sub-Families

The Base Line Range

The low cost and small size Baseline family contain PIC 10, sum PIC 10, sum PIC 12 devices. Baseline microcontrollers use a 12-bit instruction word and provide the right amount of features and options to minimize expense and get the job done right. The baseline has the simplest architecture of the 8-bit family and is therefore the easiest to use and understand. (Figure 6)

To favor hardware migrability and allow the transition to more powerful PIC models with more legs, the distribution of the functions assigned to the legs is maintained, so that connection and track layout changes are minimal. Figure 7 is a table of peripherals available on the latest 8-bit PIC® MCU devices, followed by an alphabetical list of links to the pages that describe these peripherals in detail.

The increase in the number of pins means the increase in memory capacities and in the number of peripherals and resources integrated in the device.



					Peripheral Function Focus																																							
					Intelligent Analog								Waveform Control							Timing and Measurements$^{(1)}$							Logic and Math			Safety and Monitoring			Communications					User Interface			Low Power and System Flexibility			
Product Family	Pin Count	Program Flash Memory (KB)	RAM (B)	Data EE (B)	ADC (# of bits)	Comp	HSComp	DAC (# of bits)	OPA	PRG	SlopeComp	ZCD	CCP/ECCP	10-bit PWM	16-bit PWM	COG	CWG	NCO	DSM	AngTMR	HLT (8-bit)	16-bit PWM (16-bit)	NCO (20-bit)	SMT (24-bit)	RTCC	TEMP/TS	CLC	MULT	MathACC	CRC/SCAN	HLT	WWDT	EUSART/AUSART	UART with Protocols	I^2C/SPI	USB with ACT	LIN Capable	mTouch® Sensing	HCVD	LCD	PPS	IDLE/DOZE/PMD	DMA/VI	DIA/MAP
PIC10(L)F3XX	6	384–896	64	HEF	8									✓			✓	✓					✓			✓	✓											✓						
PIC12LF1552	8	3.5	256	HEF	10																					✓									✓			✓	✓					
PIC16LF155X/6X	14–20	7–14	1024	HEF	$10^{(1)}$									✓												✓							✓		2		✓	✓	✓					
PIC16(L)F145X	14–20	14	1024	HEF	10	✓								✓			✓									✓							✓		✓	✓	✓	✓						
PIC1X(L)F157X	8–20	1.75–14	1024	HEF	10	✓		5							✓		✓					✓				✓							✓				✓	✓			✓			
PIC16(L)F153XX	8–48	3.5–28	2048	HEF	10	✓		5				✓	✓	4			✓	✓			✓		✓			✓	✓				✓	✓	2		2		✓	✓			✓	✓		✓
PIC1X(HV)F752/53	8–14	1.75–3.5	128	–	10		✓	5/9	✓		✓		✓			✓					✓										✓							✓						
PIC1X(L)F1612/3	8–14	3.5	256	HEF	10	✓		8				✓	✓				✓				✓			✓		✓				✓	✓	✓						✓						
PIC16(L)F161X	14–20	7–14	1024	HEF	10	✓		8				✓	✓	✓			✓			✓	✓			✓		✓	✓		✓	✓	✓	✓	✓		✓		✓	✓			✓			
PIC16(L)F170X	14–20	3.5–14	1024	HEF	10		✓	5/8	✓			✓	✓	✓		✓										✓	✓						✓		✓		✓	✓	✓		✓			
PIC16(L)F171X	28–40	7–28	2048	HEF	10		✓	5/8	✓			✓	✓	✓		✓		✓					✓			✓	✓						✓		✓		✓	✓	✓		✓			
PIC16(L)F176X/7X	14–40	7–28	2048	HEF	10		✓	5/10	✓	✓	✓	✓	✓	✓	✓	✓			✓		✓	✓				✓	✓				✓		✓		✓		✓	✓			✓			
PIC16(L)F183XX	8–20	3.5–14	2048	256	10	✓		5					✓	✓			✓	✓	✓				✓			✓	✓						✓		2		✓	✓			✓	✓		
PIC16(L)F184XX	14–28	7–28	2048	256	$12^{(2)}$	✓		5				✓	✓	✓			✓	✓			✓		✓	✓		✓	✓					✓	✓		2			✓	✓		✓	✓		✓
PIC16(L)F188XX	28–40	7–56	4096	256	$10^{(2)}$	✓		5				✓	✓	✓			✓	✓	✓		✓		✓	✓		✓	✓			✓	✓	✓	✓		2		✓	✓	✓		✓	✓		
PIC16(L)F191XX	28–64	14–56	4096	256	$12^{(2)}$	✓		5				✓	✓	✓			✓				✓			✓	✓	✓	✓				✓	✓	✓		✓		✓	✓	✓	$✓^{(3)}$	✓	✓		✓
PIC18(L)FXXK40	28–64	16–128	3728	256–1K	$10^{(2)}$	✓		5				✓	✓	✓			✓		✓		✓					✓		✓		✓	✓	✓	5		2		✓	✓	✓		✓	✓		
PIC18(L)FXXK42	28–48	16–128	8192	256–1K	$12^{(2)}$	✓		5				✓	✓	4			✓	✓	✓		✓		✓	✓		✓	✓	✓		✓	✓	✓	1	1	2		✓	✓	✓		✓	✓	✓	✓
PIC18(L)FXXJ94	64–100	32–128	4096	–	12	✓							✓												✓			✓					4		2	✓	✓	✓		$✓^{(3)}$	✓	✓		
PIC18(L)FXXK83	28	32–64	4096	1K	$12^{(2)}$	✓		5				✓	✓	✓			✓	✓	✓		✓		✓	✓		✓	✓	✓		✓	✓	✓		2	✓		✓	✓	✓		✓	✓	✓	✓
PIC18FXXQ10	28–40	128	3728	1024	$10^{(2)}$	✓		5				✓	✓	✓			✓		✓		✓					✓	✓			✓	✓	✓	2		2			✓	✓		✓			

Figure 7. 8-bit Peripheral Summary

The Mid Range

The microcontrollers that make up this range respond to a repertoire of 35 instructions with a format of 14 bits in length each, the Stack being 8 levels deep and having an interrupt vector. It is a large range that currently reaches 71 different models, starting with those that are encapsulated with 8 legs and reaching those with 68 legs. Figure 8 shows the most relevant aspects of some PICs of the 8-leg Medium range and others with more.

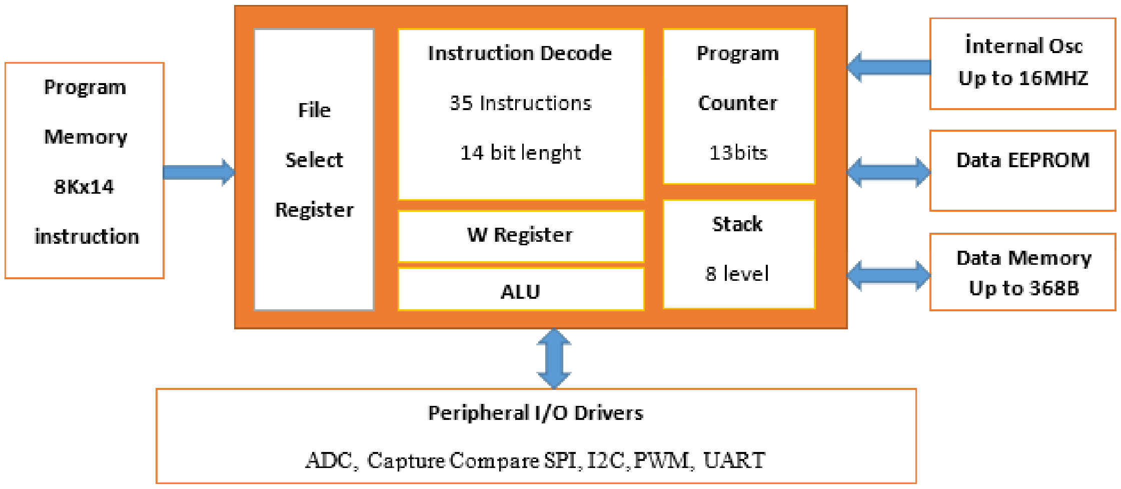


Figure 8. Mid Range Core

Enhanced Mid-Range

To improve the performance of two PIC Mid-Range microcontrollers, the PIC Enhaced Mid-Range family was raised. This family increased its performance by more than 50% compared to mid-range microcontrollers. Its core has a multiplier for Hardware and optimized for programming in C. Or code created by compiler C is reduced by 40% compared to the Mid-Range family in Figure 9. It has a lower cost than the PIC18 microcontrollers.

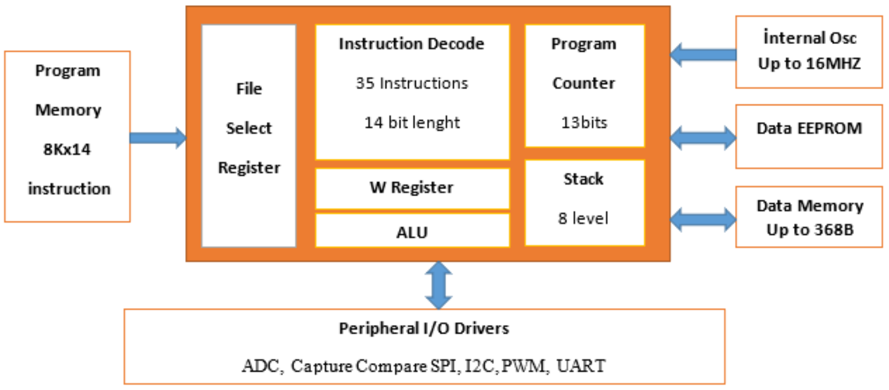


Figure 9. Enhanced Mid-Range Core

Advanced feature set, multiple serial communication and motor control capability

High-End

To the PIC18 family it presents a higher level of performance and integration of two 8-bit microcontrollers from Microchip. Possuem 16-bit instructions with more than 16 MIPS processing, 32-level hardware pile, 8 × 8 multiplier for Hardware, or that makes linguagem the best option for programming the architecture.

These microcontrollers present various advanced peripherals such as: CAN, USB, Ethernet, LCD among others.

To follow are presented its main characteristics:

- 83 instructions with optimization for programming in C language;
- I attached 2 MB program memory;
- 4KB of RAM;
- 32 snow hardware pilot;
- 8 × 8 multiplier by hardware;

The great interest of the manufacturer in this powerful model range can be felt, as it has the largest number of different devices. The capacity of program memory can reach 128 KB, data memory up to 3963 bytes and EEPROM up to 1 KB. It has highly specialized peripherals, among which a 10-bit AD converter stands out, up to 5 timers, interfaces for communication with I2C bus, SPI, USART, can 2.0 b, etc. Among the important resources is a fast hardware multiplier that allows this operation to be performed in one instruction cycle.

PIC 16F84 Architecture

Since simple applications require few resources and more complex applications require numerous and powerful resources, Microchip builds various models of microcontrollers geared to meet the needs of each project.

The architecture and programming of all PICs is very similar. In this topic we will focus on the PIC16F84 microcontroller, which is one of the mid-range microcontrollers. Its main characteristics are the following:

- Part of the data memory is EEPROM type (64 8-bit registers).
- Flash-type program memory (1024 registers of 14 bits), with the same performance as EEPROM but better performance.
- Two timers: TMR0 and watchdog. The TMR0 can act as a timer or a counter.
- Four possible interrupt sources that can be enabled or disabled by software.
- System reinitialization or RESET for five different causes.
- State of operation in low consumption or Sleep, with a consumption of 40 mA.
- Maximum working frequency can be 10 MHz.

Block Diagram

A somewhat simplified internal architecture (Fig. 10) has been derived from the block diagram given in the data sheet.This architecture applied by Microchip in its microcontrollers is characterized by the independence between code and data memory. Thus, both the capacity and the size of the buses of each memory are strictly adapted to the needs of the design, facilitating the parallel work of the two memories, which allows obtaining high levels of performance. The RISC philosophy is evident in the small number of instructions that make up its repertoire. It only consists of 35 instructions, which are executed in an instruction cycle, equivalent to four clock periods, except for jump instructions that require two cycles.

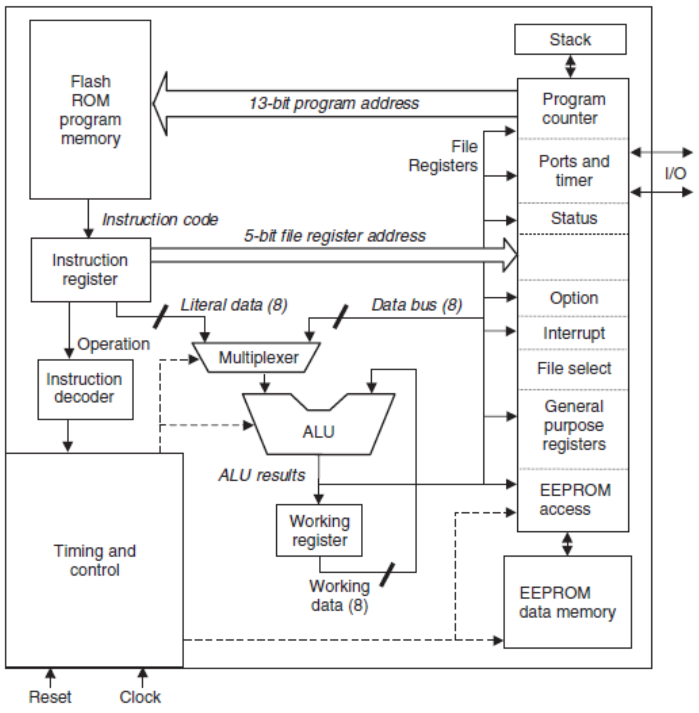


Figure 10. PIC 16F84 Internal Architecture.

The file recordset contains the port registers and the program counter, as well as various audit and status registers. The most commonly used ports (A and B), status register (status), real time clock counter (TMR0) and interrupt control (INTCON). There are also a number of redundant General Purpose registers (GPRs) that can be used as data registers, counters, etc. File records are numbered 00–4f, but appropriate tags are usually given in the program source code. File registers also provide access to an EEPROM block, which is a non-volatile data memory.

Thanks to its ports, the microcontroller can respond to external events and perform a certain action, such as varying the output signals according to the value of the inputs. To respond to external events, the microcontroller has a resource known as interrupts. Interruptions are random events, which are generated inside or outside the microcontroller, they stop the execution of the main program, to attend a secondary program or subprogram.

A real application is for example a push button connected to an input pin. Once pressed, an interrupt signal is generated, which will start the execution of the interrupt subroutine, which can be a timer, which at the end of the programmed time activates an output signal, this signal could turn off an oven for example or simply turn on a led.

To determine the input and output needs of the system, it is convenient to draw a block diagram of the system, in such a way that it is easy to identify the amount and type of signals to be controlled. Once this analysis has been carried out, it may be necessary to add peripherals, external hardware or change to another microcontroller more suitable for that system.

Ports A and B

The ports of the Microchip 16F84 microcontroller (Figure 11) are a set of pins or lines that are grouped together, each group of pins or lines makes up a complete port, the 16F84 has two ports implemented, port A and port B, both ports are bidirectional and They can be configured as input or output, in this way the ports allow the input or output of data to or from the internal part of the microcontroller, port A is made up of five pins or lines, ranging from RA0 to RA4, as It is observed in figure 1.5, pin 3 is multiplexed, that is, it fulfills a double function, which corresponds to RA4 / T0CKI, this is the only pin of port A that is multiplexed, and it can work as a RA4 port or as an input of the counter external events T0CKI, for the latter, pin 3 must be configured as an input.

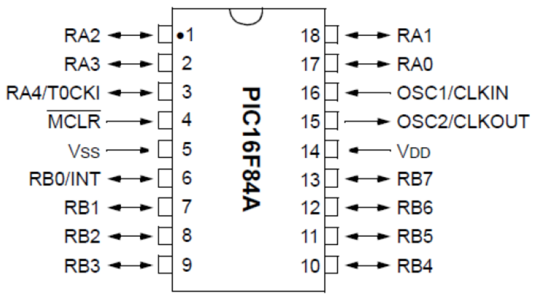


Figure 11. The Peripheral Diagram Shows in Great Detail the Function of each Pin.

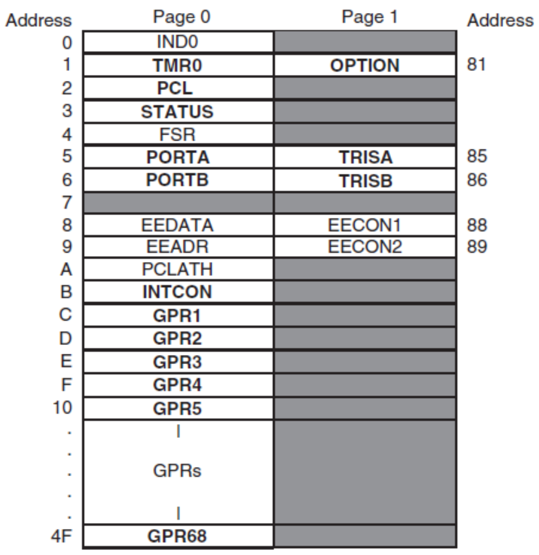
Pin 3 fulfills double function, and you cannot use both functions at the same time, through the configuration you choose which function that pin will fulfill, and if it will be input or output, the other port with more pins is port B, and it is made up of eight pins or lines, which go from RB0 to RB7, as shown in figure 1.5, pin 6 is multiplexed, that is, it fulfills a double function, which corresponds to RB0 / INT, this is the only pin or line that is multiplexed, and can work as RB0 port or as external interrupt input INT, but never as port and external interrupt input at the same time, through the configuration you choose the function that the pin or line will fulfill and if it will be input or output.

The ports A and B of the microcontroller can be configured as input or output, according to the requirements of the application. If the port is configured as an output, the pins of the port work in source mode, supply current, and the maximum current that a line or pin is 20mA. If the port is configured as input, the pins work in sink mode, they receive current, and the maximum current that can enter through a line or pin is 25mA. If in the application there are loads that consume higher currents, such as: stepper motors, relays, various LEDs and other peripherals that require more current than indicated by the manufacturer, a matching coupler circuit must be applied. Pull-up resistors must be activated every time it is necessary, and not leave pins or input lines in a floating state, in digital electronics two logical states are handled. This problem is solved by activating an internal resistance between the input pin and Vdd, called a pull-up resistor, its objective is to ensure a logic 1 when the switch is open. If what is desired is to ensure a logical 0, the resistor is connected to Vss and is called a pull-down resistor.

Registers Sets

It is necessary to indicate that there are two internal registers associated with the ports, register with port should not be confused, just as there is a port A with 5 pins, there is an internal register called PORTA, port A with the PORTA register are closely related. These registers in some cases are called words or bytes, which means the same thing, in the registers the data reading or writing operations can be carried out, applying the MOV data movement instructions.

All the file registers are 8-bits wide. Figure 12 shows PIC 16F84 file register set. They are divided into two main blocks Special Function Registers (SFR), which are reserved for specific purposes, and General Function Registers (GPR) which can be used for temporary storage of any data byte.



Address	Page 0	Page 1	Address
0	IND0	▨	
1	**TMR0**	**OPTION**	81
2	**PCL**	▨	
3	**STATUS**	▨	
4	FSR	▨	
5	**PORTA**	**TRISA**	85
6	**PORTB**	**TRISB**	86
7	▨	▨	
8	EEDATA	EECON1	88
9	EEADR	EECON2	89
A	PCLATH	▨	
B	**INTCON**	▨	
C	**GPR1**	▨	
D	**GPR2**	▨	
E	**GPR3**	▨	
F	**GPR4**	▨	
10	**GPR5**	▨	
. . . . .	GPRs	▨	
4F	**GPR68**	▨	

Figure 12. PIC 16F84 File Register Set.

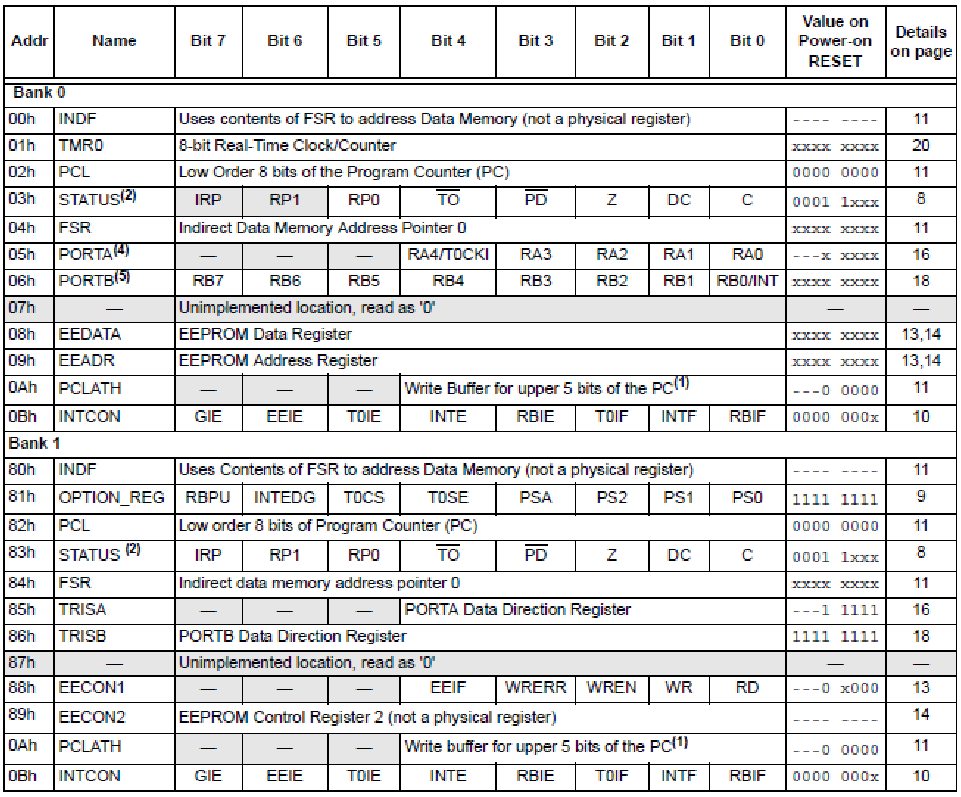
In the PIC there are many internal registers, these registers are contained in memory banks, a memory bank is nothing more than a set of registers ordered sequentially. Each register has a technical and unique name. There are microcontrollers that have two up to four memory banks, the 16F84 has two memory banks, each bank is assigned a name, bank 0 and bank 1.

In the memory banks there is a space of 68 bytes that the manufacturer left free, this space is known as RAM memory and is occupied by the program variables, this space starts at address 0x0C and ends at address 0x4F of bank 0 , according to the free space, up to 68 variables can be declared, which is a sufficient amount for application development, The procedure to address the memory banks will be explained in the following points. Figure 1.9 shows the graphic representation of the two memory banks of the PIC 16F84, which was copied from the manufacturer's manual.

Special Function Registers

The SFR Special Function Registers (Figure 12 and 13) are used by the CPU and peripheral functions to control the operation of the device. These registers are the static RAM.

The special function registers can be classified into two sets, central and peripheral. Those associated with the main functions are described in this section. Those related to the operation of peripheral features are described in the section for that specific feature.



Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on RESET	Details on page
Bank 0											
00h	INDF	Uses contents of FSR to address Data Memory (not a physical register)								---- ----	11
01h	TMR0	8-bit Real-Time Clock/Counter								xxxx xxxx	20
02h	PCL	Low Order 8 bits of the Program Counter (PC)								0000 0000	11
03h	STATUS[2]	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	8
04h	FSR	Indirect Data Memory Address Pointer 0								xxxx xxxx	11
05h	PORTA[4]	—	—	—	RA4/T0CKI	RA3	RA2	RA1	RA0	---x xxxx	16
06h	PORTB[5]	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0/INT	xxxx xxxx	18
07h	—	Unimplemented location, read as '0'								—	—
08h	EEDATA	EEPROM Data Register								xxxx xxxx	13,14
09h	EEADR	EEPROM Address Register								xxxx xxxx	13,14
0Ah	PCLATH	—	—	—	Write Buffer for upper 5 bits of the PC[1]					---0 0000	11
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	10
Bank 1											
80h	INDF	Uses Contents of FSR to address Data Memory (not a physical register)								---- ----	11
81h	OPTION_REG	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	9
82h	PCL	Low order 8 bits of Program Counter (PC)								0000 0000	11
83h	STATUS [2]	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	8
84h	FSR	Indirect data memory address pointer 0								xxxx xxxx	11
85h	TRISA	—	—	—	PORTA Data Direction Register					---1 1111	16
86h	TRISB	PORTB Data Direction Register								1111 1111	18
87h	—	Unimplemented location, read as '0'								—	—
88h	EECON1	—	—	—	EEIF	WRERR	WREN	WR	RD	---0 x000	13
89h	EECON2	EEPROM Control Register 2 (not a physical register)								---- ----	14
0Ah	PCLATH	—	—	—	Write buffer for upper 5 bits of the PC[1]					---0 0000	11
0Bh	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	10

Figure 13. Special Function Register File Summary

Interrupts on PIC 16F84:

Mid-range PICs are capable of handling 4 sources of interrupts:

- External (RB0 / INT pin).
- A change in bits RB4 to RB7 of Port B.
- Timer interruption (terminal count).

- Completion of writing to the EEPROM memory.
- There is a single interrupt vector at address 0004h.
- The INTCON register (position 0Bh, both banks) allows each interrupt to be enabled (equivalent to individual masks).
- This same register contains flags that serve to indicate (to the service subroutine) which was the origin of the interruption. INTCON also has a general enablement bit (GIE) that affects all sources.

Mid-Range Instruction Set

Special features of the assembler for PIC

- Uses the mnemonics of the instructions of the µC to which it is directed. In our case, it will correspond to the instruction set of the mid-range PICs (only 35 instructions).
- Allows the use of labels, both to point to addresses and to define constant values.
- It is possible to express numerical quantities in different bases, the default base being hexadecimal.
- There are a series of pseudo-instructions or directives (in addition to the well-known ORG, EQU and END) specific to these µCs.

Each PIC16CXX instruction is a 14-bit word divided into an opcode that indicates the type of instruction and one or more operands that further indicate the instruction's execution. The pic16cxx instruction set summary in Figure 13 lists byte-oriented, bit-oriented, and literal and control operations. Figure 14 shows the opcode field descriptions.

To better understand each instruction, the meaning of some parameters will be explained below:

f: Register affected by the instruction

w: Accumulator (Working register)

b: Bit number (there are instructions that affect a single bit)

k: Constant (a number)

d: Selection of the destination of the result of the instruction, it can be "0" or "1", if it is "0" the result is saved in the accumulator (w) and if it is "1" it is saved in the register f to which affects instruction.

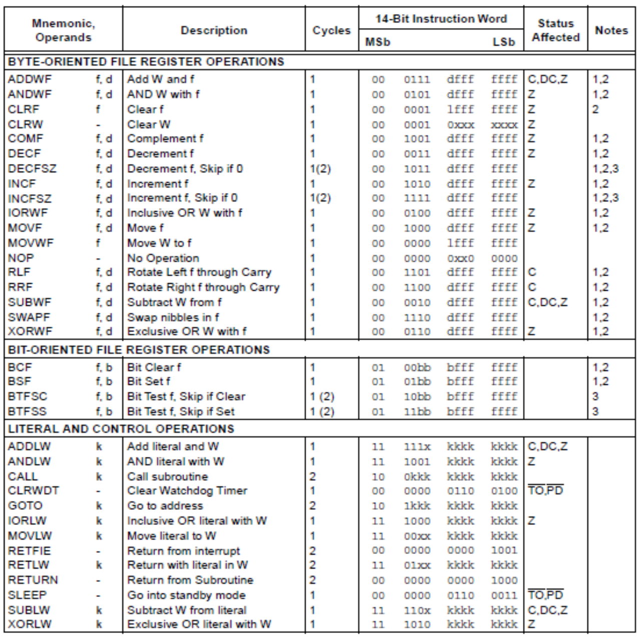
Data Transfer Instructions

Data transfer within the microcontroller takes place between the working register and a register that represents anywhere in the internal RAM, whether it relates to special function or general purpose registers.

Move the first three instruction literals to the W register, move the data from the W register to RAM and from RAM to the W register. The CLRF instruction clears the f register, while the CLRW clears the W register. The swap instruction changes nibbles within the F register.

Arithmetic-Logic Instructions

Similar to most microcontrollers, the PIC only supports two arithmetic instructions - addition and subtraction. Flags C, DC, z are automatically set depending on the results of addition or subtraction. The only exception is the C flag. The C flag is inverted after it is subtracted, since the subtraction is performed by an addition with a negative value. This means that the C flag is set if it is possible to perform the operation and cleared if the larger number is subtracted from the smaller one. The logic can perform one (1) operation of the PIC and or, EX-or, inversion (COMF) and rotation (RLF and RRF). Instructions that return a register actually rotate their bits one bit left (toward bit 7) or right (toward bit 0) along the C flag. The bit shifted from the register is automatically moved to the C flag which is moved to the bit on the opposite side of the register.



Mnemonic, Operands		Description	Cycles	14-Bit Instruction Word				Status Affected	Notes
				MSb			LSb		
BYTE-ORIENTED FILE REGISTER OPERATIONS									
ADDWF	f, d	Add W and f	1	00	0111	dfff	ffff	C,DC,Z	1,2
ANDWF	f, d	AND W with f	1	00	0101	dfff	ffff	Z	1,2
CLRF	f	Clear f	1	00	0001	1fff	ffff	Z	2
CLRW	-	Clear W	1	00	0001	0xxx	xxxx	Z	
COMF	f, d	Complement f	1	00	1001	dfff	ffff	Z	1,2
DECF	f, d	Decrement f	1	00	0011	dfff	ffff	Z	1,2
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00	1011	dfff	ffff		1,2,3
INCF	f, d	Increment f	1	00	1010	dfff	ffff	Z	1,2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00	1111	dfff	ffff		1,2,3
IORWF	f, d	Inclusive OR W with f	1	00	0100	dfff	ffff	Z	1,2
MOVF	f, d	Move f	1	00	1000	dfff	ffff	Z	1,2
MOVWF	f	Move W to f	1	00	0000	1fff	ffff		
NOP	-	No Operation	1	00	0000	0xx0	0000		
RLF	f, d	Rotate Left f through Carry	1	00	1101	dfff	ffff	C	1,2
RRF	f, d	Rotate Right f through Carry	1	00	1100	dfff	ffff	C	1,2
SUBWF	f, d	Subtract W from f	1	00	0010	dfff	ffff	C,DC,Z	1,2
SWAPF	f, d	Swap nibbles in f	1	00	1110	dfff	ffff		1,2
XORWF	f, d	Exclusive OR W with f	1	00	0110	dfff	ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS									
BCF	f, b	Bit Clear f	1	01	00bb	bfff	ffff		1,2
BSF	f, b	Bit Set f	1	01	01bb	bfff	ffff		1,2
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01	10bb	bfff	ffff		3
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01	11bb	bfff	ffff		3
LITERAL AND CONTROL OPERATIONS									
ADDLW	k	Add literal and W	1	11	111x	kkkk	kkkk	C,DC,Z	
ANDLW	k	AND literal with W	1	11	1001	kkkk	kkkk	Z	
CALL	k	Call subroutine	2	10	0kkk	kkkk	kkkk		
CLRWDT	-	Clear Watchdog Timer	1	00	0000	0110	0100	$\overline{TO}$,$\overline{PD}$	
GOTO	k	Go to address	2	10	1kkk	kkkk	kkkk		
IORLW	k	Inclusive OR literal with W	1	11	1000	kkkk	kkkk	Z	
MOVLW	k	Move literal to W	1	11	00xx	kkkk	kkkk		
RETFIE	-	Return from interrupt	2	00	0000	0000	1001		
RETLW	k	Return with literal in W	2	11	01xx	kkkk	kkkk		
RETURN	-	Return from Subroutine	2	00	0000	0000	1000		
SLEEP	-	Go into standby mode	1	00	0000	0110	0011	$\overline{TO}$,$\overline{PD}$	
SUBLW	k	Subtract W from literal	1	11	110x	kkkk	kkkk	C,DC,Z	
XORLW	k	Exclusive OR literal with W	1	11	1010	kkkk	kkkk	Z	

Figure 14. Midrange Instruction Set

Bit-oriented Instructions

BCF and BSF instructions clear or set any bits in memory. Although it may seem like a simple process, it is not. The CPU first reads the entire byte, changes a bit, and rewrites the entire byte to the same location.

Typical fixed connection for any application.

In the circuits where the PIC16F84 is used, it is usual to use a supply voltage of + 5V and as an external clock circuit one of the XT type at a frequency of 4MHz. With this configuration, the fixed wiring for any application is shown in Figure 15. The pins that are not connected are those dedicated to transferring the information with the peripherals used by the application.

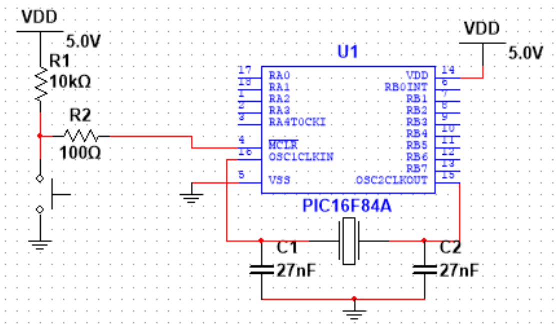


Figure 15. Typical Fixed Connection for any Application.

Actual designs use various peripherals that must be connected to the microcontroller pins that support the I / O lines.

Pushbuttons.

These devices allow a logic level to be entered at the moment they are activated, passing to the opposite level when they are stopped (they return to the rest position).

In the diagram on the left of Figure 16 the input line receives a high logic level when the pushbutton is idle and a low logic level when it is actuated. The button on the right works in reverse.

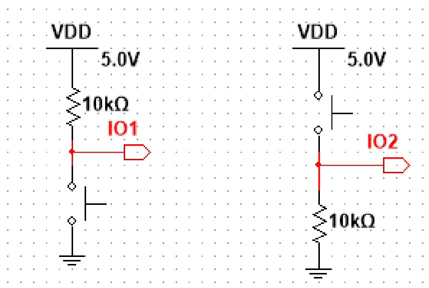


Figure 16. Two Possible Ways to Connect a Pushbutton.

There are a multitude of detectors, limit switches and digital sensors that work in the same way as push buttons.

LEDs.

The led diode is an element that is used as a light indicator. When the potential difference between its anode and its cathode exceeds a certain threshold value, the led diode will turn on. The PIC lines can supply enough current to turn on an LED diode, so they can be connected directly through a resistor as shown in Figure 17. If we use the connection on the left of the figure, the led diode will light when the microcontroller output is set to ‘1’, while with the connection on the right it will do so when the output is set to ‘0’.

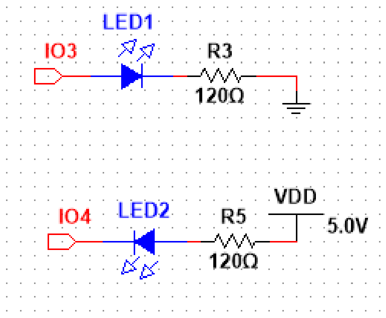


Figure 17. LED Connections

Diodes or other loads sometimes require more current than the PIC lines can deliver. In this case it is necessary to insert an amplifier stage.

Relays

Activating and deactivating a relay provides the opportunity to control much larger loads (more current) because they can be controlled by the relay contacts (Figure 18).

When the output line, OUT, applies a high level to the base of the Darlington transistor (amplifier stage) causing it to conduct and activate the relay. Closing the relay contacts controls a higher load. The value of the resistance depends on the type of relay and the transistor.

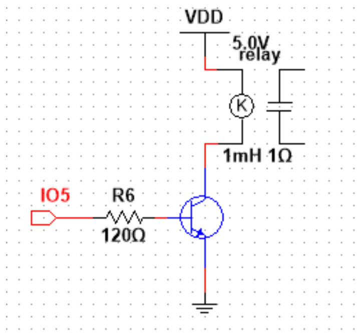


Figure 18. Scheme of the Control of a Relay.

General Structure of an Assembler Program.

In a program written in assembly language, in addition to the 35 instructions interpreted by the processor, directives are also placed, which are commands to perform certain operations with the program. The parts that generally exist in a program are discussed below:

1. Processor model and numbering system.

Programs start with the list directive that references the microcontroller model. The type of numbering to be used with the radix directive is also usually specified. The usual thing is to use the hexadecimal system, in which the values are expressed preceded by "0x".

List p = 16F84; PIC16F84 microcontroller is used

2. Variables

Data memory locations are used to store operands and results, as well as to store special registers.

To make it easier for the programmer to make the program, instead of referring to the memory locations where the data to be used is located, a name is associated with each of these locations. The equ directive relates a name to the address that is assigned, so the programmer works with names and the compiler automatically translates these to the corresponding addresses. For example, the register that contains the status information is at address 0x03, input port A at 0x05, etc. If we want to use variable names for these memory addresses, we would write:

STATUS equ 0x03 ; The label "STATUS" is associated with the address 0x03

PORT equ 0x05 ; The label "PORT" is associated with the address 0x05

3. Origin of the program.

Before starting to write machine instructions, you must define the address of the program memory from which you want to start loading the program. The org directive is used for this. In PICs, the origin of the program is always set at address 0x00 because that is where the program starts executing after doing a reset. We will define the origin as follows:

```
org 0x00                ; Program start
```

When the program handles interrupts, it does not start to load the program from address 0x00, because if an interruption is generated, the program that attends it starts at address 0x04 (interrupt vector). In this case, what is usually done is to put at address 0x00 a jump to an address of the program memory after the reset vector, for example we would jump to a position labeled START which is at address 0x05.

```
        org 0x00                    ; The following instruction will be at the beginning
        of memory

        goto START                  ; Jumps to the address labeled HOME

        org 0x05                    ; The next instruction will be at address 0x05

START

        -------

        -------

        End
```

4. Body of the program and end.

After indicating the address where the program will begin to load, follow the body of the program composed of the machine instructions and their operands.

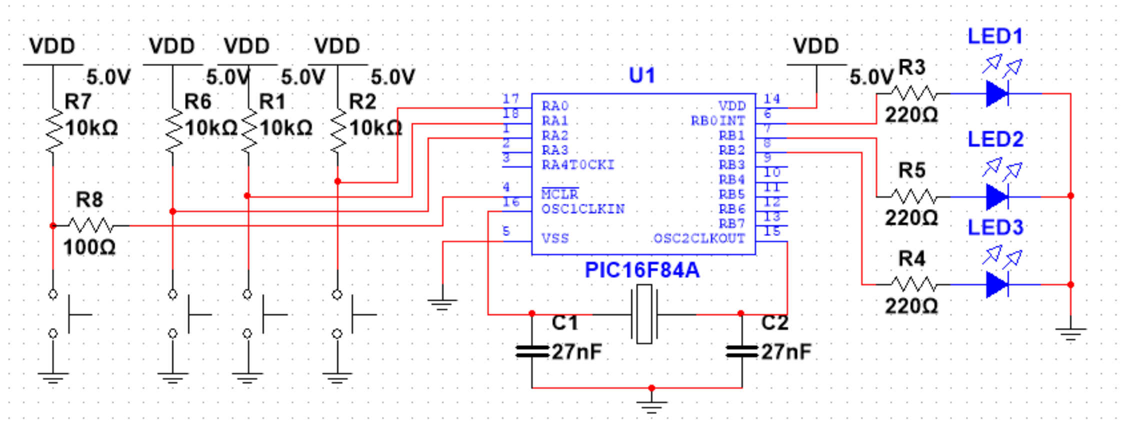


Figure 19. Sample PIC 16F84 Microcontroller

The PIC16F84 has two doors, PORT A (RA0-RA4) and PORT B (RB0-RB7). Each of the lines of these doors can act as an entrance or an exit. The value of the data entering or leaving each port is materialized in two positions in RAM, PORTA for port A and PORTB for port B (figure 6.21). There are two other registers in the RAM memory (TRISA and TRISB) that are used to configure the PORTA and PORTB lines as input or output, in such a way that when a '1' is recorded in one of the bits, the corresponding line of the affected gate acts as an input, whereas if a '0' is recorded, it acts as an output.

In positions 5 and 6 (Figure 6-22) of bank 0 the data registers of the doors are located and in the same positions of the memory, but in bank 1, there are the corresponding configuration registers. If you want to access bank 1 positions, set bit 5 (RP0) of a register called STATUS to ‘1’.

For example, if you wanted to configure all the lines of gate A as inputs and those of gate B as outputs, you would have to address bank 1 (bit 5 of STATUS = ‘1’), load the TRISA bits with ‘1’ and with ‘0’ all TRISB bits. Once the gates have been configured, bit 5 of the STATUS register will have to be set back to ‘0’ (return to bank 0) to be able to read the information entered by the lines configured as inputs or send through the lines configured as outputs.

```
List p = 16f84
STATUS equ 0x03
PORTA equ 0x05
PORTB equ 0x06
org 0x00
goto HOME
org 0x05                ; skip interrupt vector
BEGINNING
bsf STATUS, 5                   ; switch to bank1
movlw b’00000000 ’              ; W ¬‘ 00 ’
movwf PORTB                     ; TRISB¬W (PORTB outputs)
movlw b’00011111 ’              ; W ¬‘ 1F ’
movwf PORTA                     ; TRISA¬W (PORTA inputs)
bcf STATUS, 5                   ; switch to bank0
LOOP
movf PORTA, W                   ; W ¬ PORTA
movwf PORTB                     ; PORTB ¬ W
goto LOOP
end
```

References

Wilmshurst, T. (2006).*Designing embedded systems with PIC microcontrollers: principles and applications*. Elsevier.

Predko, M. (2002).*Programming and customizing PICmicro microcontrollers* (Vol. 2). Upper Saddle River, NJ: McGraw-Hill.

Bates, M. (2011). *PIC microcontrollers: an introduction to microelectronics*. Elsevier.

Peatman, J. B. (1998). *Design with PIC microcontrollers*. Pearson Education India.

Sanchez, J., & Canton, M. P. (2018). *Microcontroller programming: the microchip PIC*. CRC press.

Borowik, B. (2011).*Interfacing PIC microcontrollers to peripherial devices* (Vol. 49). Springer Science & Business Media.

Smith, D. W. (2006). *PIC in practice: a project-based approach*. Elsevier.

Ibrahim, D. (2011). *Pic Basic Projects: 30 Projects Using Pic Basic and Pic Basic Pro*. Elsevier.

Di Jasio, L. (2007). *Programming 16-bit PIC Microcontrollers in C: Learning to Fly the PIC 24*. Elsevier.

Verle, M. (2008). *PIC microcontrollers* (pp. 36-39). Mikroelektronika.

Iovine, J. (2002). *PIC microcontroller project book*. McGraw-Hill Professional.

Shea, J. J. (2005). Programming the PIC microcontroller with Mbasic [Book Review]. *IEEE Electrical Insulation Magazine*, *21*(6), 44-45.

Katzen, S. (2006). *The Quintessential PIC® Microcontroller*. Springer Science & Business Media.

Siegesmund, M. (2014). Embedded C programming: Techniques and applications of C and PIC McUs. Newnes.

Predko, M. (2002). *Programming and customizing PICmicro microcontrollers* (Vol. 2). Upper Saddle River, NJ: McGraw-Hill.

[URL1] www.mikroe.com

[URL2] https://www.microchip.com/

[URL3] https://www.statista.com/statistics/891620/microchip-technology-net-sales-by-product-line/

[URL4] https://microchipdeveloper.com/8bit:summary

[URL5] https://www.icinsights.com/news/bulletins/MCUs-Sales-To-Reach RecordHigh -Annual-Revenues-Through-2022/

[URL6] https://ww1.microchip.com/downloads/en/DeviceDoc/30009630M.pdf

About the Authors

Sabri KOCER, PhD, He graduated from the Electrical Engineering Department of Selcuk University. He completed his graduate and his doctorate in Gazi University. Currently, Necmettin Erbakan University, Faculty of Engineering, Computer Engineering is working. Electronics, Computer, Telecommunication, Signal Processing and Biomedical studies in the area.

Email: skocer@erbakan.edu.tr , Orcid: 0000-0002-4849-747X

Resul BUTUNER, is a Computer Teacher at Adil Karaagac Vocational and Technical Anatolian High School in Konya, Turkey. He has a master’s degree in Computer Engineering from Necmettin Erbakan University. His main areas of interest are artificial intelligence, robotic coding, data mining and augmented reality applications. He is an instructor in the field of Robotic coding within TUBITAK. He continues to write a book in the field of robotic coding at the Ministry of National Education. He worked as a coordinator in budgeted projects related to student education.

E-mail: rbutuner@gmail. com, Orcid: 0000-0002-9778-2349

Similarity Index

The similarity index obtained from the plagiarism software for this book chapter is 19%.

To Cite This Chapter:

Kocer, S. & Butuner, R. (2021). Pic Microcontroller. In S. Kocer, O. Dundar & R. Butuner (Eds .), *Programmable Smart Microcontroller Cards* (pp. 126 –148). ISRES Publishing.

Copyright © 2021 by ISRES Publishing