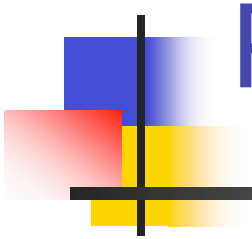
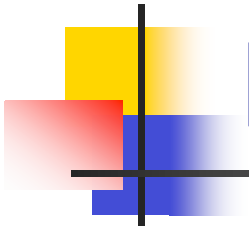


PIC Microcontroller - 1

Introduction to PIC Microcontroller

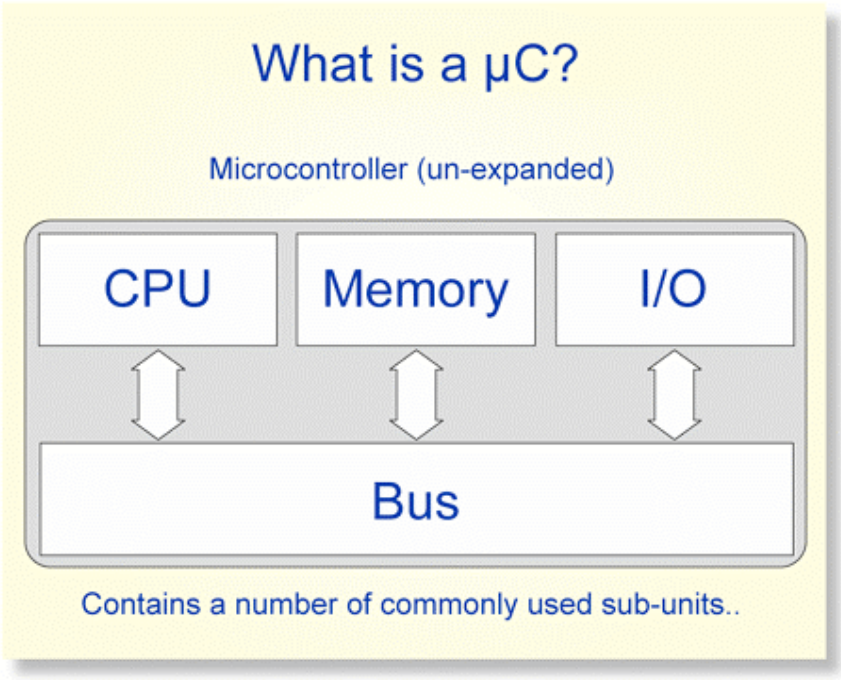
Moving Towards Embedded Hardware

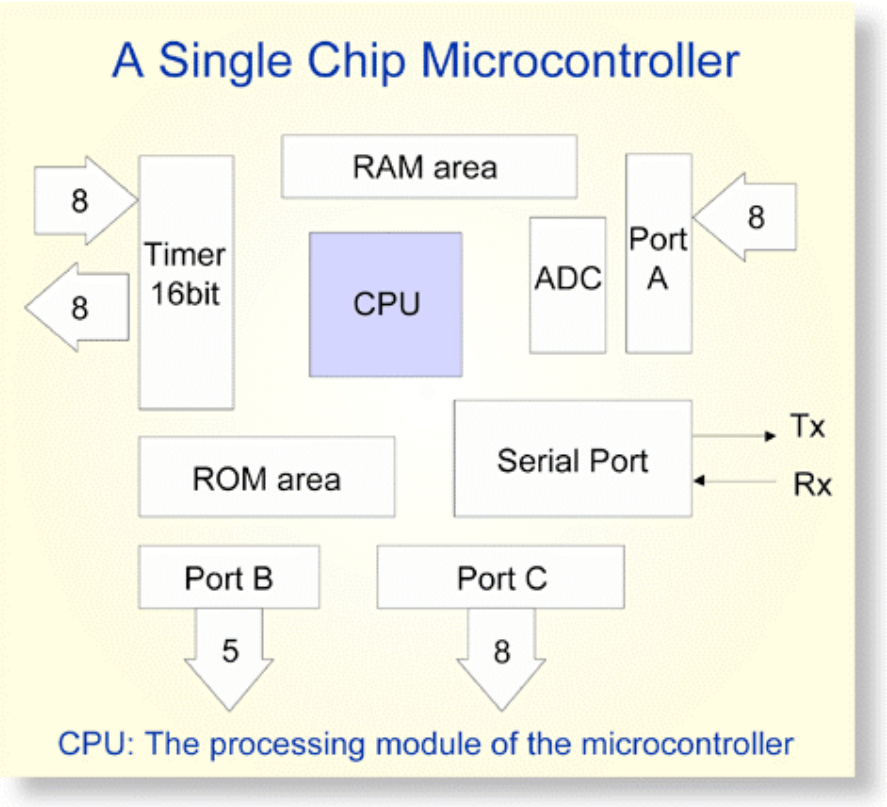


Typical components of a PC:

- x86 family microprocessor
- Megabytes or more of main (volatile) memory
- Gigabytes or more of magnetic memory (disk)
- Operating System (w/ user interface)
- Serial I/O
- Parallel I/O
- Real-time clock
- Keyboard
- Video Display
- Sound card
- Mouse
- NIC (Network Interface Card)
- etc.

Overview of Microcontrollers

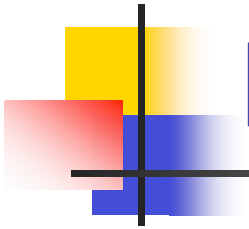




- Basically, a microcontroller is a device which integrates a number of the components of a microprocessor system onto a single microchip.

Reference: http://mic.unn.ac.uk/miclearning/modules/micros/ch1/micro01notes.html#1.4

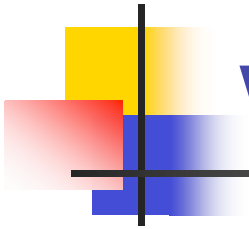
The PIC 16F87x Series Microcontroller



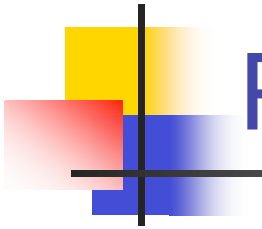
Some of the characteristics of the PIC 16F87x series

- High performance, low cost, for embedded applications
- Only 35 instructions
- Each instruction is exactly one word long and is executed in 1 cycle (except branches which take two cycles)
- 4K words (14bit) flash program memory
- 192 bytes data memory (RAM) + 128 bytes EEPROM data memory
- Eight level deep hardware stack
- Internal A/D converter, serial port, digital I/O ports, timers, and more!

What these changes mean when programming?



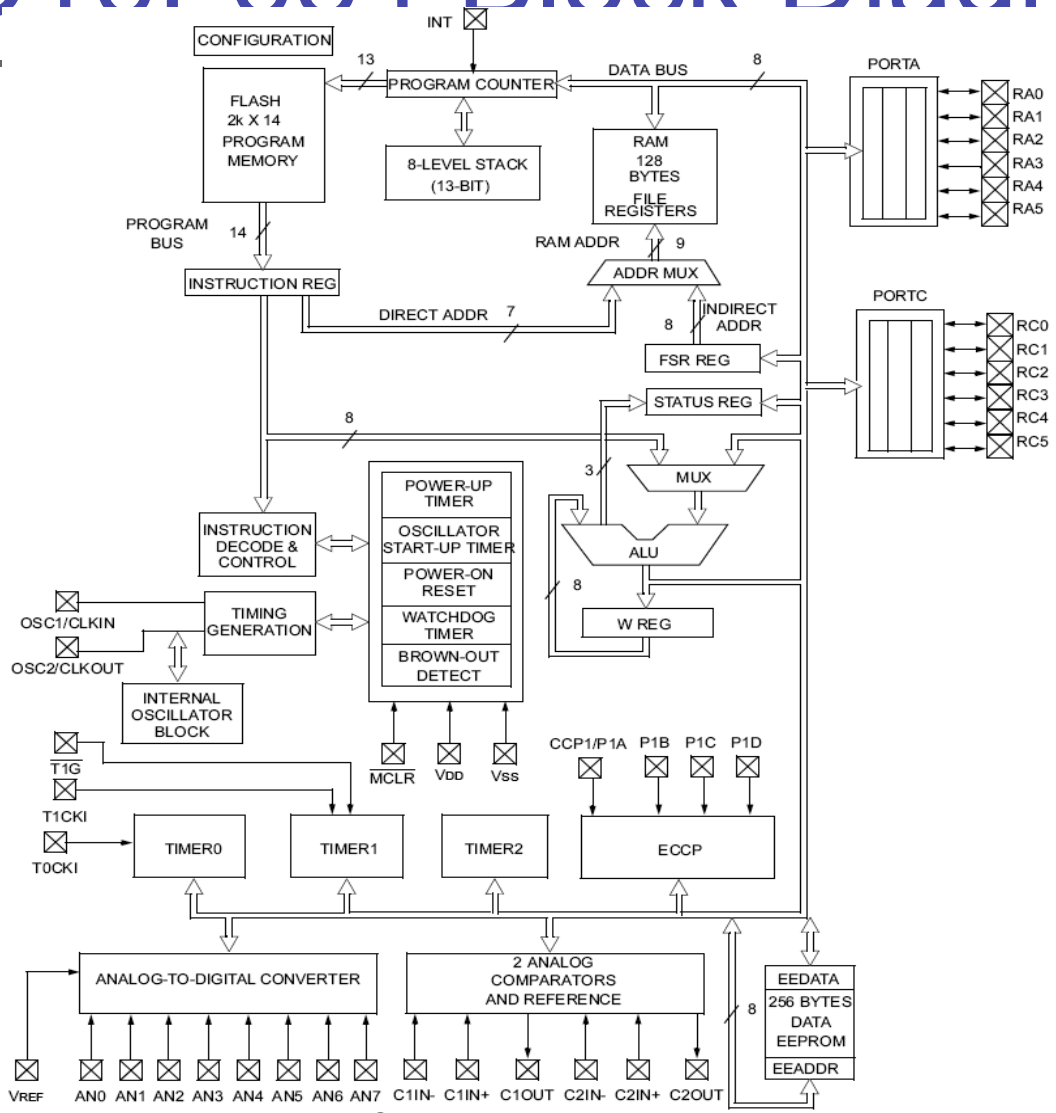
- Since there is no DOS or BIOS, you'll have to write I/O functions yourself. Everything is done directly in assembly language.
- Code design, test, and debug will have to be done in parallel and then integrated after the hardware is debugged.
- Space matters! Limitations on memory may mean being very clever about algorithms and code design.



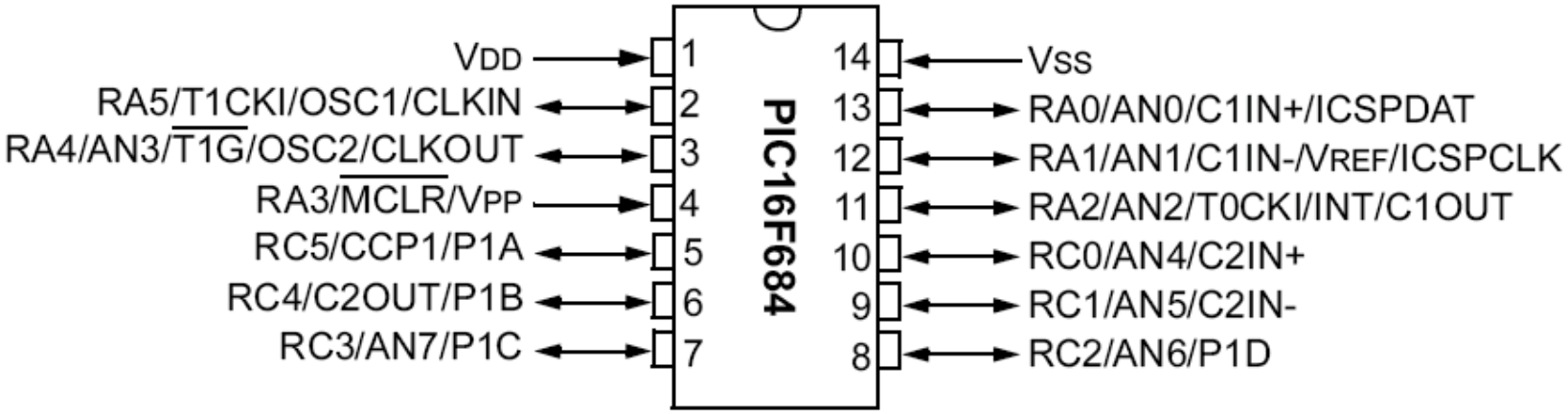
PIC Microcontroller (PIC16F684)

- High performance, low cost, for embedded applications
  - Only 35 different instructions
  - Interrupt capability
  - Direct, indirect, relative addressing mode
- Low Power
  - 8.5uA @ 32KHz, 2.0V
- Peripheral Features
  - 12 I/O pins with individual direction control
  - 10-bit A/D converter
  - 8/16-bit timer/counter
- Special Microcontroller Features
  - Internal/external oscillator
  - Power saving sleep mode
  - High Endurance Flash/EEPROM cell
  - Etc.

PIC16F684 Block Diagram



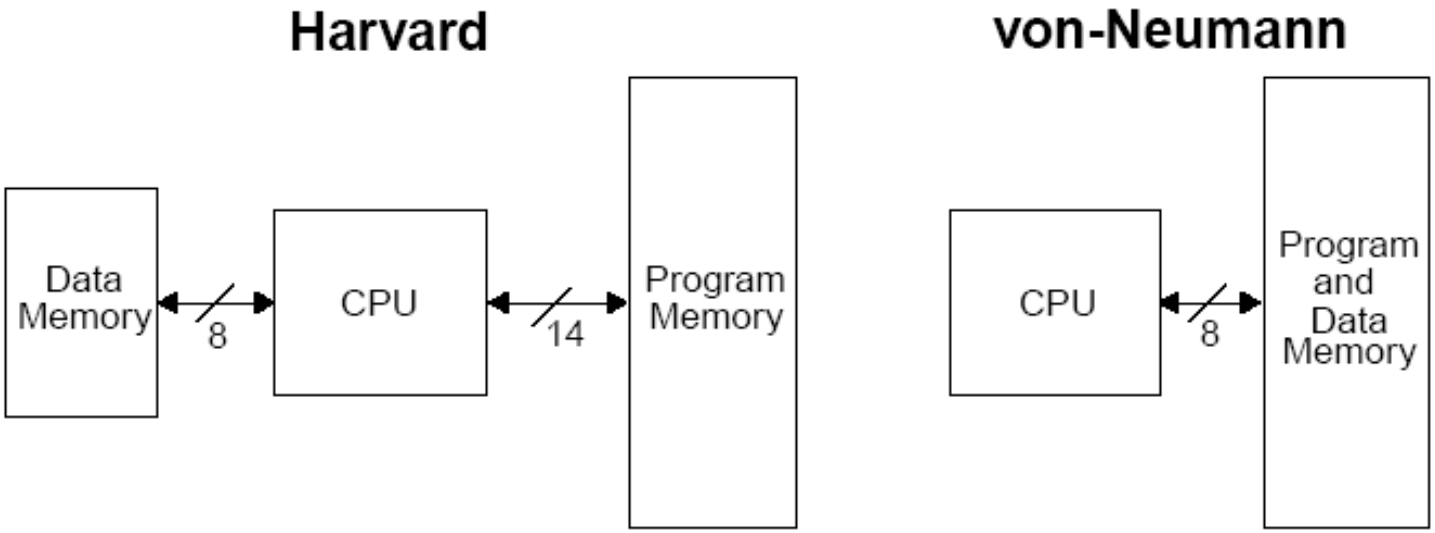
PIC16F684



- 12 pins, 2048 instructions, 128 byte variable memory, ADC, comparator, Timers, ICD

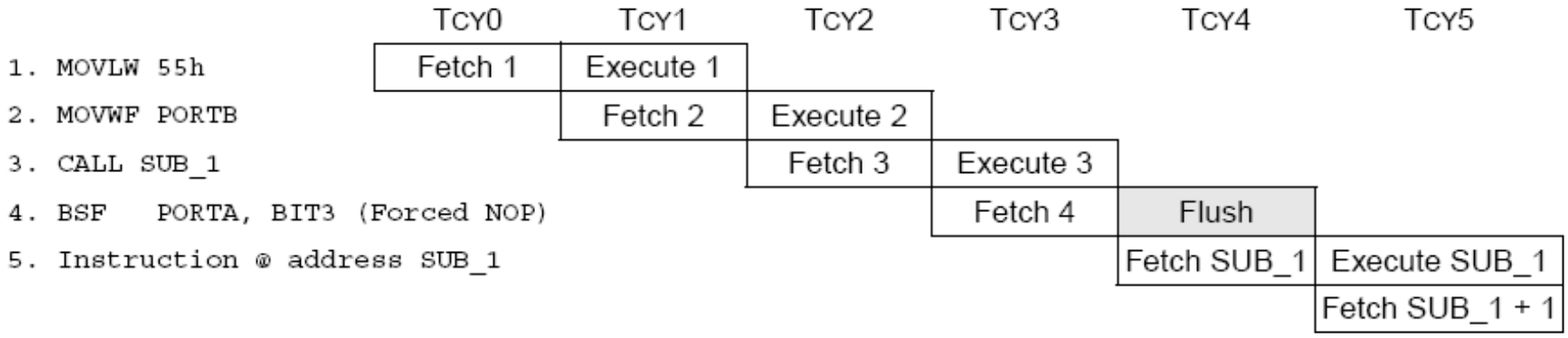
Harvard vs Von Neumann

- Organization of program and data memory



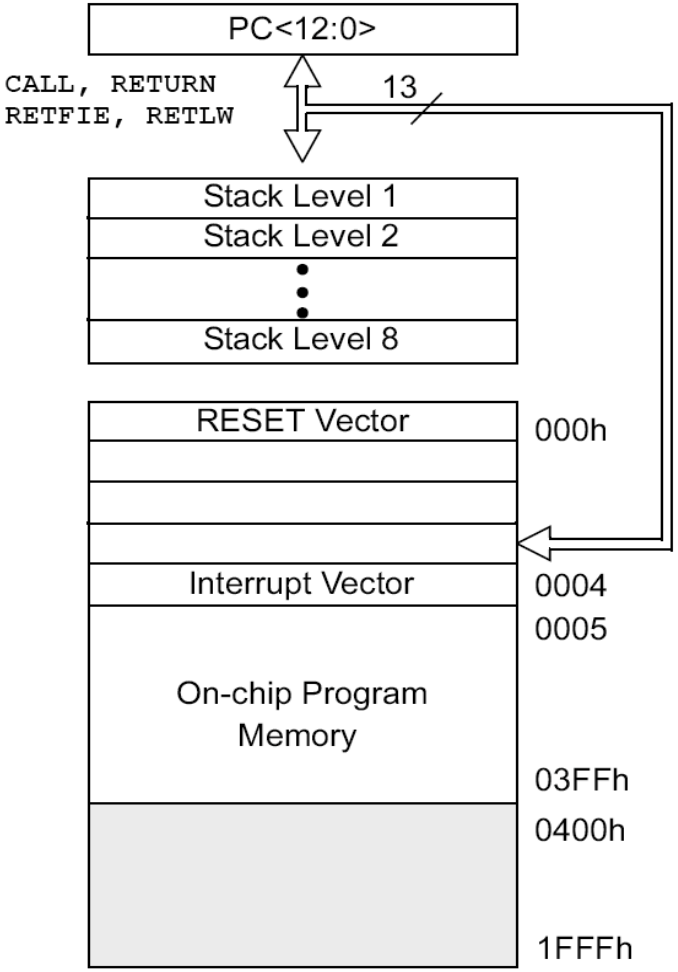
Instruction Pipelining

- It takes one cycle to fetch the instruction and another cycle to decode and execute the instruction
- Each instruction effectively executes in one cycle
- An instruction that causes the PC to change requires two cycles.



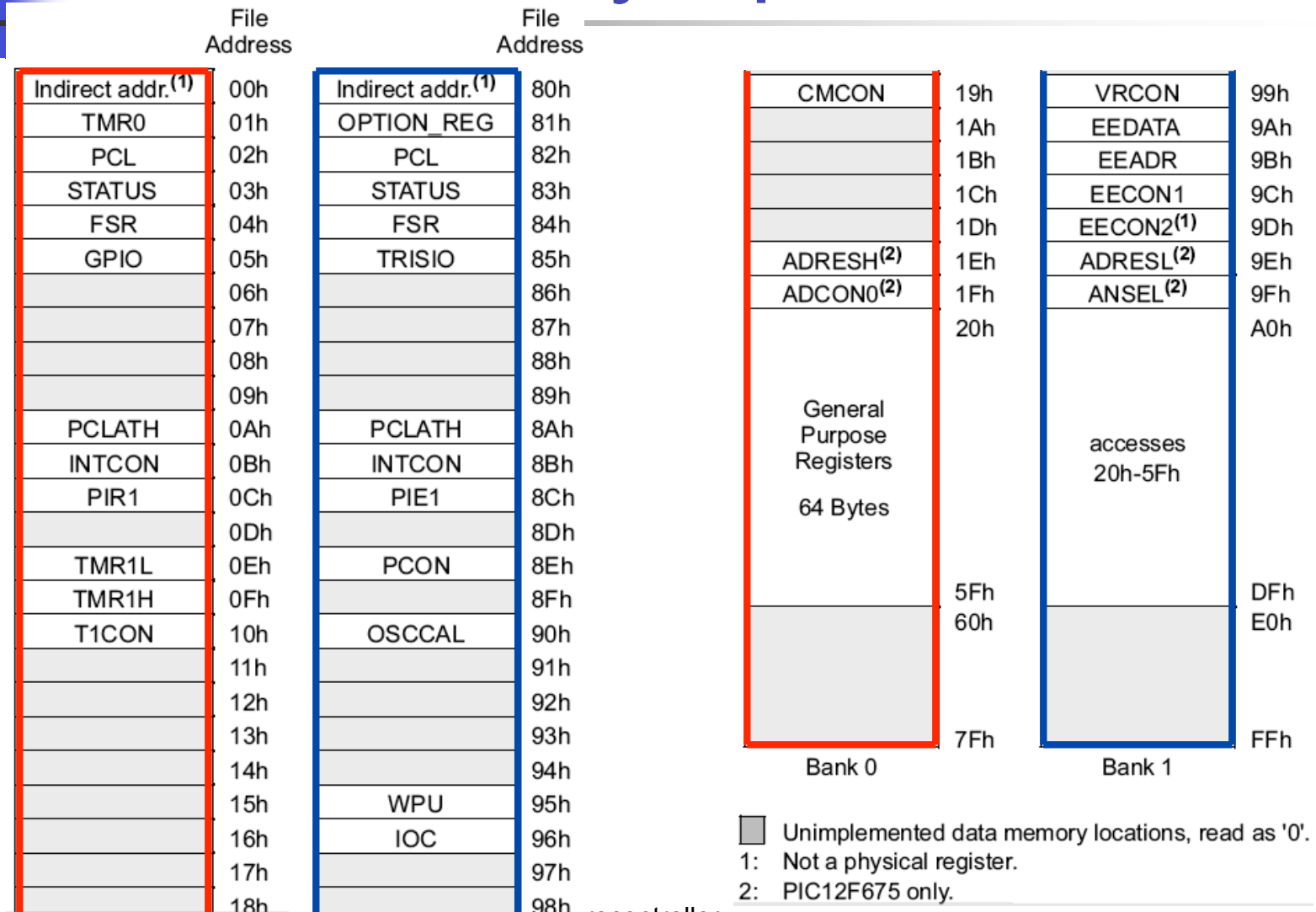
All instructions are single cycle, except for any program branches. These take two cycles since the fetch instruction is "flushed" from the pipeline while the new instruction is being fetched and then executed.

Program Memory Space



- 13-bit program counter to address 8K locations (only first 2K is implemented in 16F684)
- PIC16F675 has only 1K (x14 bit) program memory - the upper bit is simply ignored during fetches from program memory
- Each location is 14-bit wide (instructions are 14 bits long)
- RESET vector is 0000h
  - When the CPU is reset, its PC is automatically cleared to zero.
- Interrupt Vector is 0004h
  - 0004h is automatically loaded into the program counter when an interrupt occurs

Data Memory Space



Two Special addresses

- Reset Vector Address (0000h)
  - When the CPU starts up from its reset state, its PC is automatically cleared to zero.
  - Assign ***goto Mainline*** instruction

```
Mainline
    call        Initial         ;Initialize everything
MainLoop
    call        Task1           ;Deal with task1
    call        Task2           ;Deal with task2
    …
    call        LoopTime        ;Force looptime to a fixed value
    goto        MainLoop        ;repeat
```

- Interrupt Vector Address (0004h)
  - 0004h is automatically loaded into the program counter when an interrupt occurs.
  - Assign ***goto IntService*** instruction there: cause the CPU to jump to the beginning of the interrupt service routine, located elsewhere in the memory space.

Special Function Registers

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOD
Bank 0										
00h	INDF(1)	Addressing this Location uses Contents of FSR to Address Data Memory								0000 0000
01h	TMR0	Timer0 Module's Register								xxxx xxxx
02h	PCL	Program Counter's (PC) Least Significant Byte								0000 0000
03h	STATUS	IRP(2)	RP1(2)	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx
04h	FSR	Indirect Data Memory Address Pointer								xxxx xxxx
05h	GPIO	—	—	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0	--xx xxxx
0Ah	PCLATH	—	—	—	Write Buffer for Upper 5 bits of Program Counter					---0 0000
0Bh	INTCON	GIE	PEIE	T0IE	INTE	GPIE	T0IF	INTF	GPIF	0000 0000
0Ch	PIR1	EEIF	ADIF	—	—	CMIF	—	—	TMR1IF	00-- 0--0
0Eh	TMR1L	Holding Register for the Least Significant Byte of the 16-bit Timer1								xxxx xxxx
0Fh	TMR1H	Holding Register for the Most Significant Byte of the 16-bit Timer1								xxxx xxxx
10h	T1CON	—	TMR1GE	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	-000 0000
19h	CMCON	—	COUT	—	CINV	CIS	CM2	CM1	CM0	-0-0 0000
1Eh	ADRESH(3)	Most Significant 8 bits of the Left Shifted A/D Result or 2 bits of the Right Shifted Result								xxxx xxxx
1Fh	ADCON0(3)	ADFM	VCFG	—	—	CHS1	CHS0	GO/$\overline{DONE}$	ADON	00-- 0000

Status Register

Reserved	Reserved	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
bit 7							bit 0

bit 7 **IRP:** This bit is reserved and should be maintained as '0'

bit 6 **RP1:** This bit is reserved and should be maintained as '0'

bit 5 **RP0:** Register Bank Select bit (used for direct addressing)
0 = Bank 0 (00h - 7Fh)
1 = Bank 1 (80h - FFh)

bit 4 **$\overline{TO}$:** Time-out bit
1 = After power-up, CLRWDT instruction, or SLEEP instruction
0 = A WDT time-out occurred

bit 3 **$\overline{PD}$**: Power-down bit
1 = After power-up or by the CLRWDT instruction
0 = By execution of the SLEEP instruction

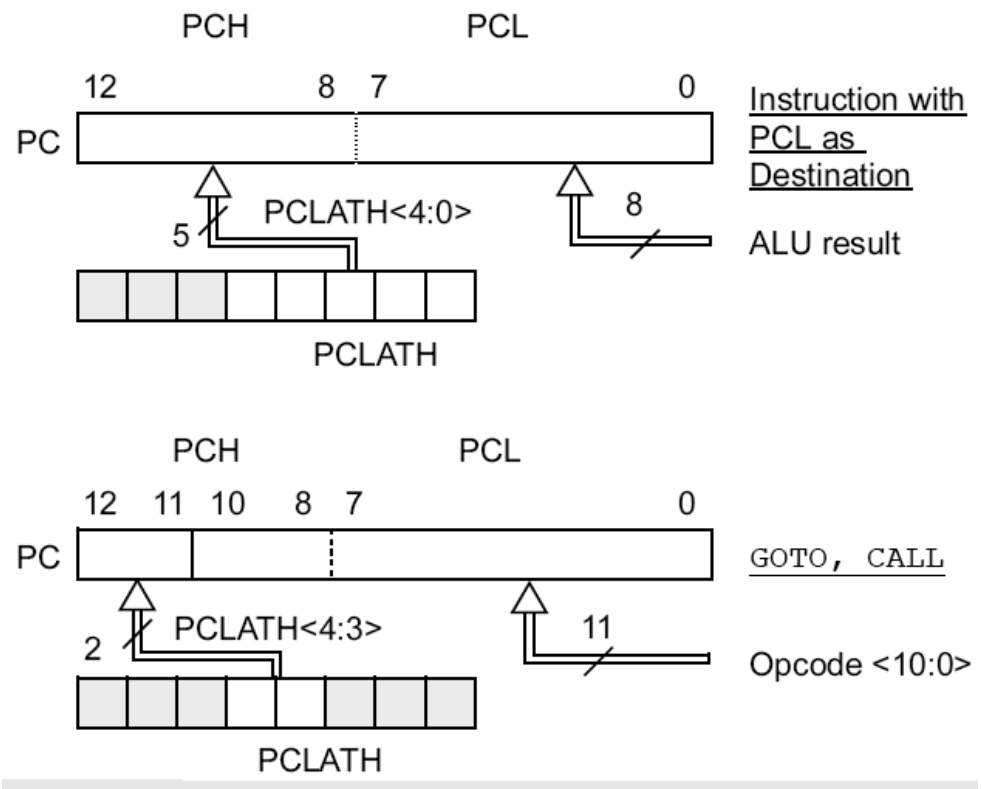
bit 2 **Z**: Zero bit
1 = The result of an arithmetic or logic operation is zero
0 = The result of an arithmetic or logic operation is not zero

bit 1 **DC**: Digit carry/$\overline{\text{borrow}}$ bit (ADDWF, ADDLW, SUBLW, SUBWF instructions)
For $\overline{\text{borrow}}$, the polarity is reversed.
1 = A carry-out from the 4th low order bit of the result occurred
0 = No carry-out from the 4th low order bit of the result

bit 0 **C**: Carry/$\overline{\text{borrow}}$ bit (ADDWF, ADDLW, SUBLW, SUBWF instructions)
1 = A carry-out from the Most Significant bit of the result occurred
0 = No carry-out from the Most Significant bit of the result occurred

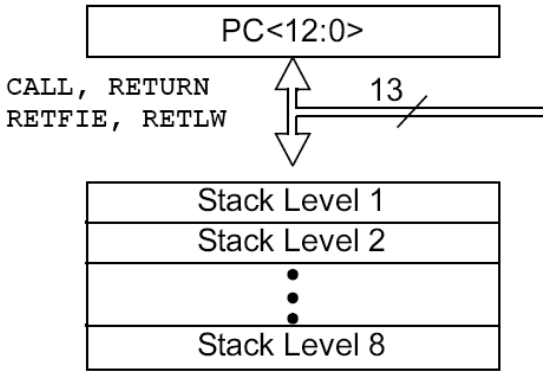
Note: For $\overline{\text{borrow}}$ the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (RRF, RLF) instructions, this bit is loaded with either the high or low order bit of the source register

PCL and PCLATH

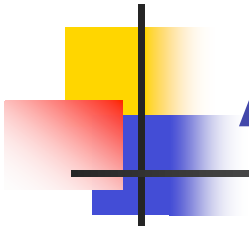


- PC: Program Counter, 13 bits
- PCL (02h): 8 bits, the lower 8 bits of PC
- PCLATH (0Ah): PC Latch, provides the upper 5 (or 2) bits of PC when PCL is written to
- 1st example: PC is loaded by writing to PCL
- 2nd example: PC is loaded during a CALL or GOTO instruction
- PCLATH is used to for jumping across 2K pages (thus, not needed in PIC16F684)

Stack



- 8-level deep x 13-bit wide hardware stack
- The stack space is not part of either program or data space and the stackpointer is not readable or writable.
- The PC is “PUSHed” onto the stack when a CALL instruction is executed, or an interrupt causes a branch.
- The stack is “POPed” in the event of a RETURN, RETLW or a RETFIE instruction execution.
- However, NO PUSH or POP instructions !
- PCLATH is not affected by a “PUSH” or “POP” operation.
- The stack operates as a circular buffer:
  - after the stack has been PUSHed eight times, the ninth push overwrites the value that was stored from the first push.



Acknowledgement

Some slides are revised based on lecture notes used in WPI ECE 2801