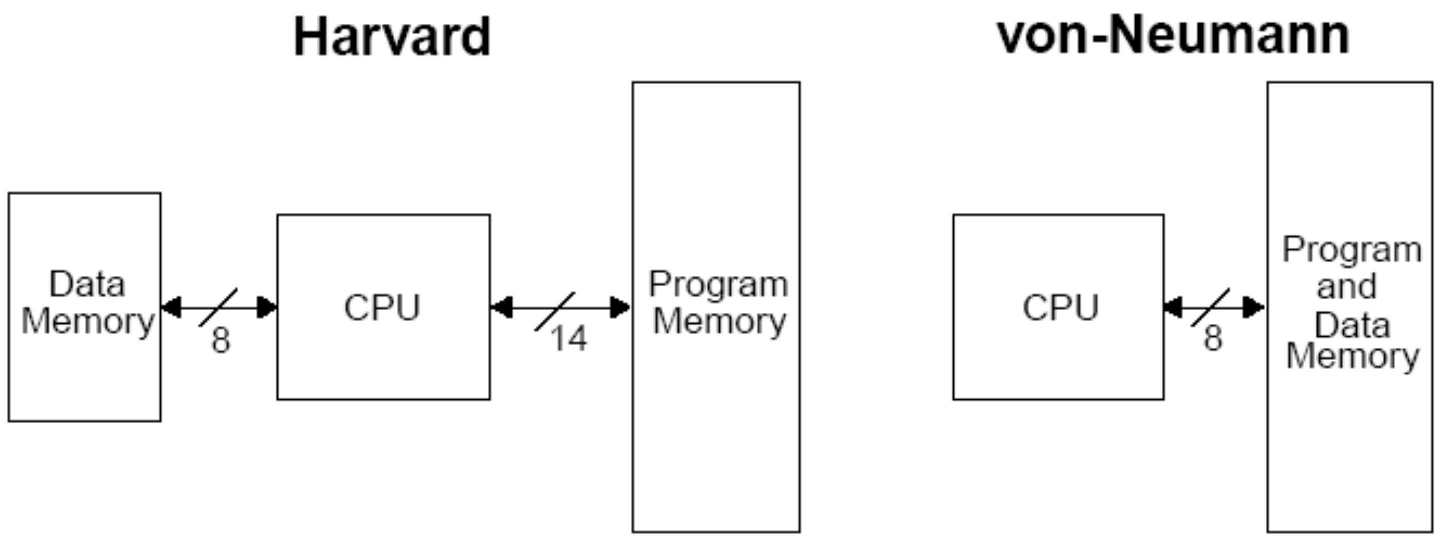
PIC Microcontroller - 2

Data Memory of PIC Microcontroller

Harvard vs Von-Neumann

- Organization of program and data memory

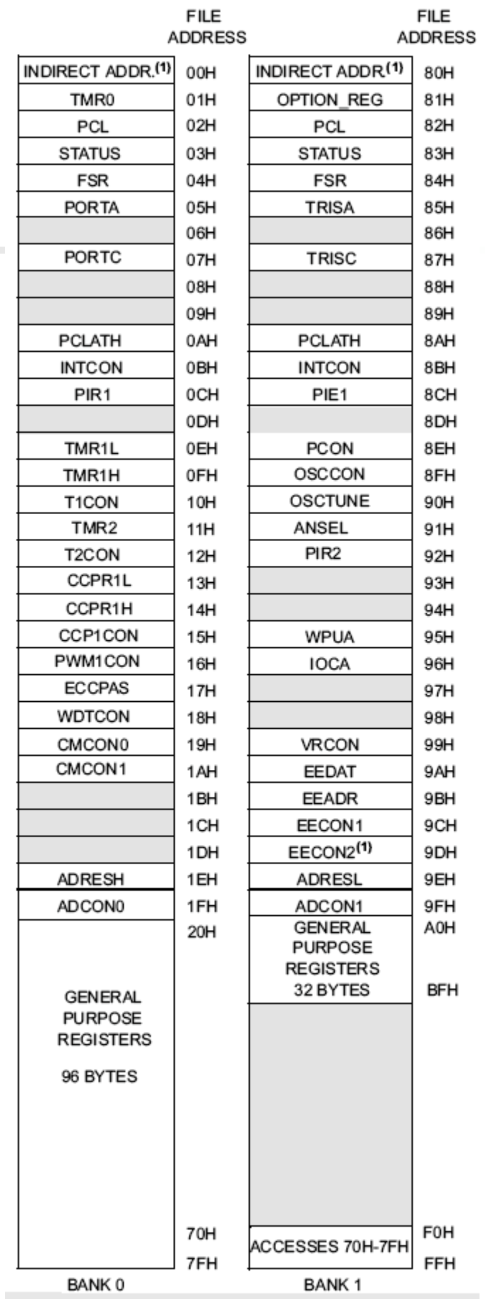


Review

- Harvard architecture
  - Separated Program memory and Data memory, separated access
- Instruction pipelining
  - While the first instruction is executed (which might involves data memory access), the second one is fetched from the program memory
- Program memory
  - Up to 8K words (13 bit PC, 2^{13}=8K)
  - Each word is 14 bits
  - Each instruction is exactly 1 word long.
  - General program structure – two special addresses: Reset vector address (0000h) and Interrupt vector address (0004h)
  - 8 level deep hardware stack

Data Memory Map

- Data memory consists of
  - Special Function Registers (SFR) area
  - General Purpose Registers (GPR) area
- SFRs control the operation of the device
- GPRs are area for data storage and scratch pad operations
- GPRs are at higher address than SFRs in a bank
- Different PIC microcontrollers may have different number of GPRs



BANK 0	FILE ADDRESS	BANK 1	FILE ADDRESS
INDIRECT ADDR.[1]	00H	INDIRECT ADDR.[1]	80H
TMR0	01H	OPTION_REG	81H
PCL	02H	PCL	82H
STATUS	03H	STATUS	83H
FSR	04H	FSR	84H
PORTA	05H	TRISA	85H
	06H		86H
PORTC	07H	TRISC	87H
	08H		88H
	09H		89H
PCLATH	0AH	PCLATH	8AH
INTCON	0BH	INTCON	8BH
PIR1	0CH	PIE1	8CH
	0DH		8DH
TMR1L	0EH	PCON	8EH
TMR1H	0FH	OSCCON	8FH
T1CON	10H	OSCTUNE	90H
TMR2	11H	ANSEL	91H
T2CON	12H	PIR2	92H
CCPR1L	13H		93H
CCPR1H	14H		94H
CCP1CON	15H	WPUA	95H
PWM1CON	16H	IOCA	96H
ECCPAS	17H		97H
WDTCON	18H		98H
CMCON0	19H	VRCON	99H
CMCON1	1AH	EEDAT	9AH
	1BH	EEADR	9BH
	1CH	EECON1	9CH
	1DH	EECON2[1]	9DH
ADRESH	1EH	ADRESL	9EH
ADCON0	1FH	ADCON1	9FH
GENERAL PURPOSE REGISTERS 96 BYTES	20H	GENERAL PURPOSE REGISTERS 32 BYTES	A0H
			BFH
	70H		F0H
	7FH	ACCESSES 70H-7FH	FFH

Banking

- Data memory is partitioned into banks
- Each bank extends up to 7Fh (128) bytes
  - 4 banks : 4*128 bytes = 512 bytes
  - 2 banks : 2*128 bytes = 256 bytes
- Lower locations of each bank are reserved for SFRs. Above the SFRs are GPRs.
- Implemented as Static RAM
- Some “high use” SFRs from bank0 are mirrored in the other banks (e.g., INDF, PCL, STATUS, FSR, PCLATH, INTCON)
- RP0 and RP1 bits in the STATUS register selects the bank when using direct addressing mode.

What are the two/four banks for?

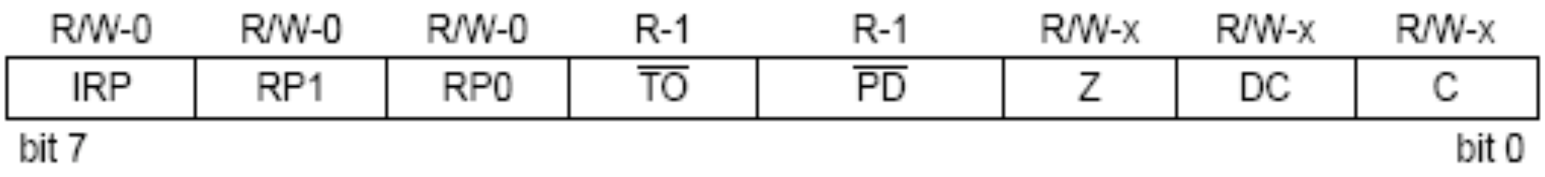
- 14-bit instructions use 7 bits to address a location
- Memory space is organized in 128Byte banks.
- PIC 16F684 has two banks - Bank 0 and Bank 1.
- Bank 1 is used to control the actual operation of the PIC
  - for example to tell the PIC which bits of Port A are input and which are output.
- Bank 0 is used to manipulate the data.
- An example is as follows: Let us say we want to make one bit on Port A high.
  - First we need to go to Bank 1 to set the particular bit, or pin, on Port A as an output.
  - We then come back to Bank 0 and send a logic 1 (bit 1) to that pin.

Special Function Registers (1)

- W, the working register
  - To move values from one register to another register, the value must pass through the W register.
- FSR (04h,84h,104h,184h), File Select Register
  - Indirect data memory addressing pointer
- INDF (00h,80h,100h,180h)
  - accessing INDF accesses the location pointed by IRP+FSR
- PC, the Program Counter, PCL (02h, 82h, 102h, 182h) and PCLATH (0Ah, 8Ah, 10Ah, 18Ah)

Special Function Registers (2)

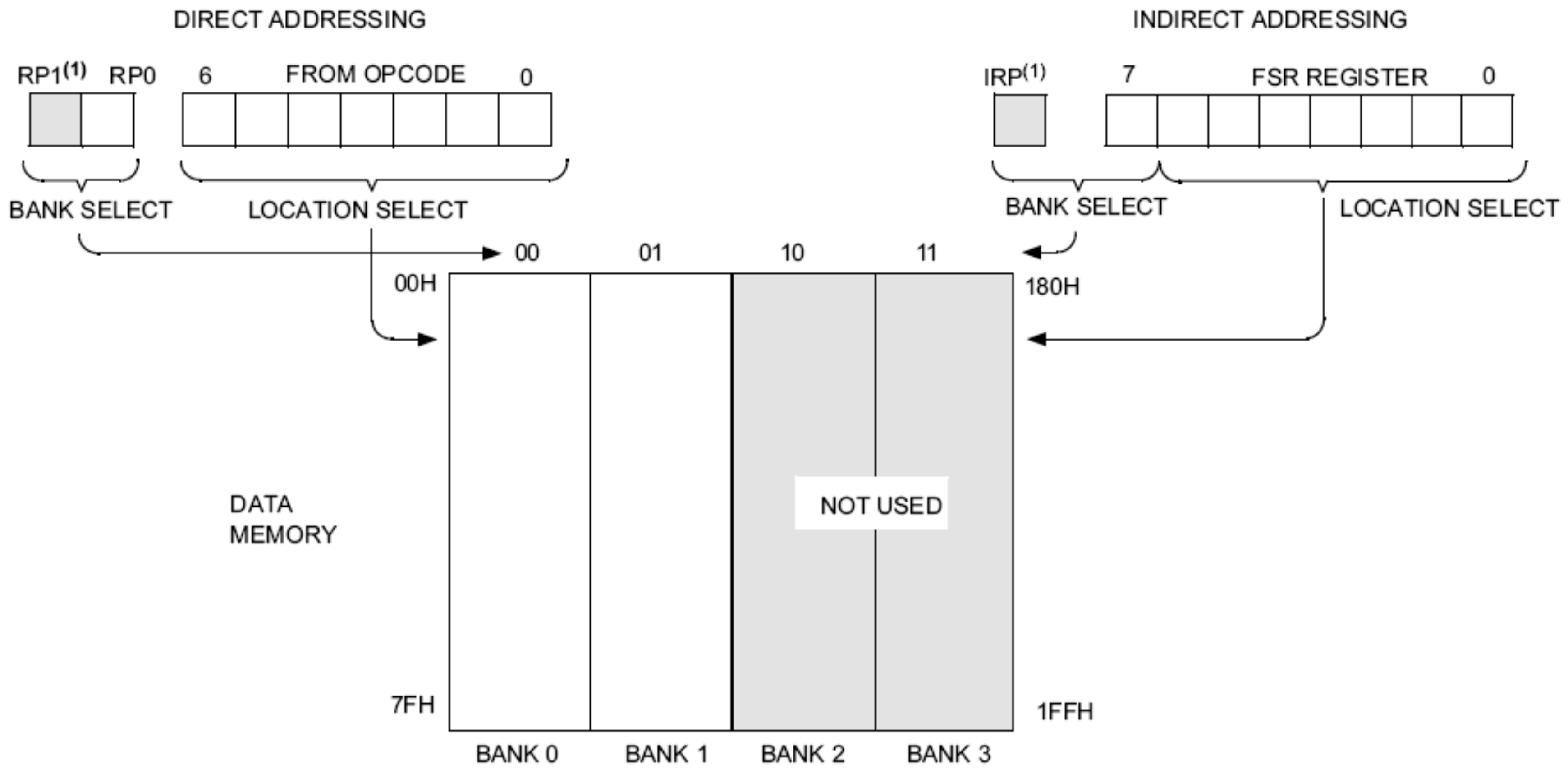
- STATUS (03h, 83h, 103h, 183h)



R/W-0	R/W-0	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

  - IRP: Register bank select bit (indirect addressing)
  - RP1:RP0 – Register bank select bits (direct addressing)
  - NOT_TO: Time Out bit, reset status bit
  - NOT_PD: Powel-Down bit, reset status bit
  - Z: Zero bit ~ ZF in x86
  - DC: Digital Carry bit ~ AF in x86
  - C: Carry bit ~ CF in x86 (note: for subtraction, borrow is opposite)

Direct/Indirect Addressing



For memory map detail see Figure 2-2.

Note 1: The RP1 and IRP bits are reserved; always maintain these bits clear.

Direct Addressing

- Use only 7 bits of instruction to identify a register file address
- The other two bits of register address come from RP0 and RP1 bits in the STATUS register

Accessed Bank	Direct (RP1:RP0)	Indirect (IRP)
0	0 0	0
1	0 1	
2	1 0	1
3	1 1	

Example: Bank switching (Note: case of 4 banks)

```
    CLRF      STATUS          ; Clear STATUS register (Bank0)
    : ;
    BSF       STATUS, RP0     ; Bank1
    : ;
    BCF       STATUS, RP0     ; Bank0
    : ;
    MOVLW     0x60            ; Set RP0 and RP1 in STATUS register, other
    XORWF     STATUS, F                 ; bits unchanged (Bank3)
    : ;
    BCF       STATUS, RP0     ; Bank2
    : ;
    BCF       STATUS, RP1     ; Bank0
```

Indirect Addressing

- The INDF register is not a physical register. Addressing the INDF register will cause indirect addressing.
- Any instruction using the INDF register actually access the register pointed to by the File Select Register (FSR).
- The effective 9-bit address is obtained by concatenating the 8-bit FSR register and the IRP bit in STATUS register.

Example

```
            MOVLW    0x20       ;initialize pointer
            MOVWF    FSR        ;to RAM
NEXT:       CLRF     INDF       ;clear INDF register
            INCF     FSR,F      ;inc pointer
            BTFSS    FSR,4      ;all done?  (to 0x2F)
            GOTO     NEXT       ;no clear next
CONTINUE
            :                   ;yes continue
```

I/O Ports

- General I/O pins are the simplest of peripherals used to monitor and control other devices.
- For most ports, the I/O pin's direction (input or output) is controlled by the data direction register **TRISx** (x=A,B,C,D,E): a '1' in the TRIS bit corresponds to that pin being an input, while a '0' corresponds to that pin being an output
- The **PORTx** register is the latch for the data to be output. Reading PORTx register read the status of the pins, whereas writing to it will write to the port latch.
- Example: Initializing PORTD (PORTD is an 8-bit port. Each pin is individually configurable as an input or output).

```
bcf     STATUS, RP0   ; bank0
bcf     STATUS, RP1
clrf    PORTD          ; initializing PORTD by clearing output data latches
bsf     STATUS, RP0   ; select bank1
movlw   0xCF           ; value used to initialize data direction
movwf   TRISD          ; PORTD<7:6>=inputs, PORTD<5:4>=outputs,
                       ; PORTD<3:0>=inputs
```

Example (Page 174)

Acknowledgement

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