CpE 213
Digital Systems Design

Lecture 4

Tuesday 9/4/2003

Announcements

- Course documents have been uploaded to Blackboard
- Lecture notes will remain on the web page for one week after being posted
- Check later today for an assignment
- Today's office hours have been moved to tomorrow from 1:30 to 3:00 pm
- Regular office hours will resume next week

Extra credit opportunity

- Get 5 (out of total 100 points)
- Come to my office (EECH219) during office hours to have your picture taken
- Deadline: Thursday 9/11 at 3pm

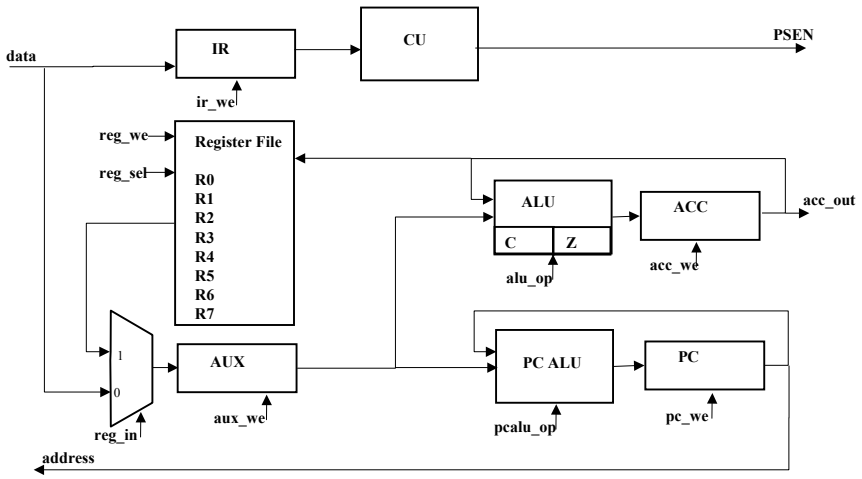
Before the next lecture

- Review WIMP handout
- Select a spokesperson from your group
- The spokesperson for each group should email me the names of all group members
- Deadline for this is tomorrow at 5pm
- Download assignment 1 (due on 9/16)

Question

- Why are D flip-flops more commonly used than JK in microcontroller circuits?

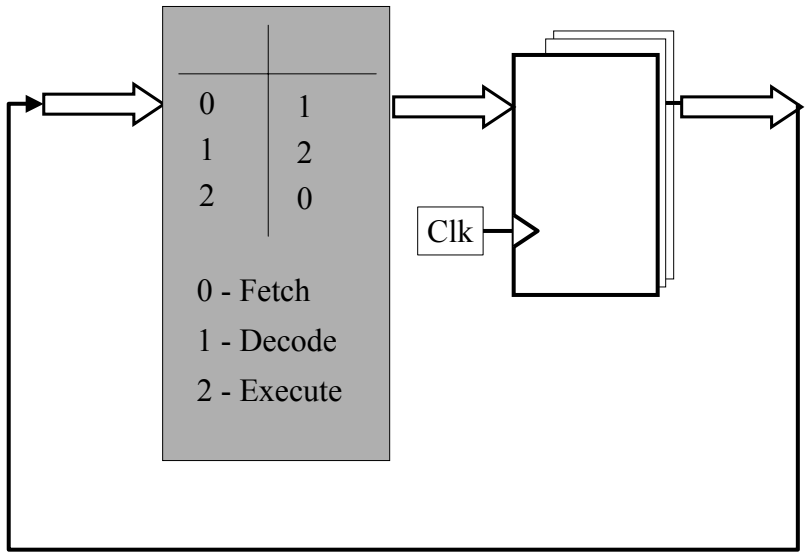
Wimp51 Microprocessor



Wimp51 control unit

- Questions
  - Draw a timing diagram showing a clock, the current state, and next state for at least two instruction cycles.
  - Describe what needs to be done to implement the state machine controller in hardware.

Wimp51 control unit



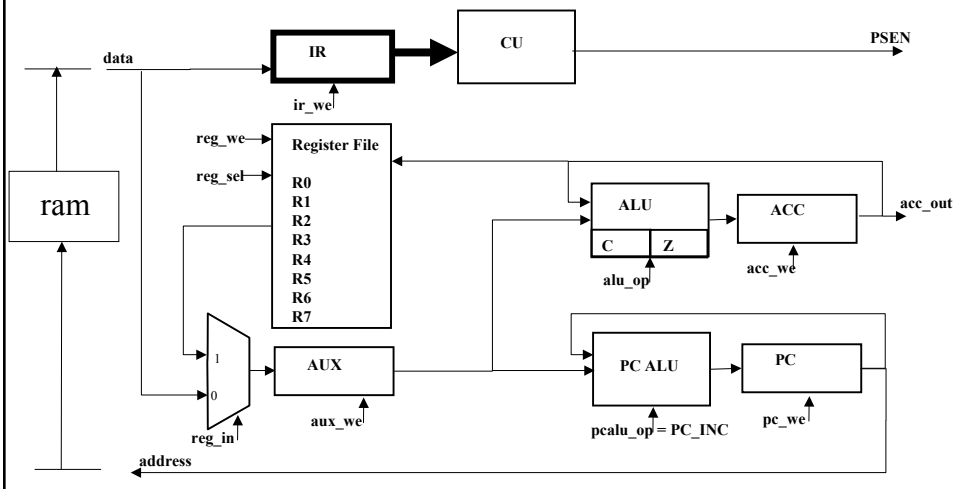
Questions

- What is the range of code addresses in the WIMP51?
- What is the range of memory addresses (in hexadecimal)?

Question

- A D-FF will latch its input whenever a rising clock edge occurs. We only want IR to latch the data bus at the end of the fetch cycle and not during the other two states. How can we cause IR to change at the end of the decode cycle and keep it the same during the other two cycles? This is an important technique which we'll want to use on all the other machine registers.

Decode cycle: Decode current instruction in IR

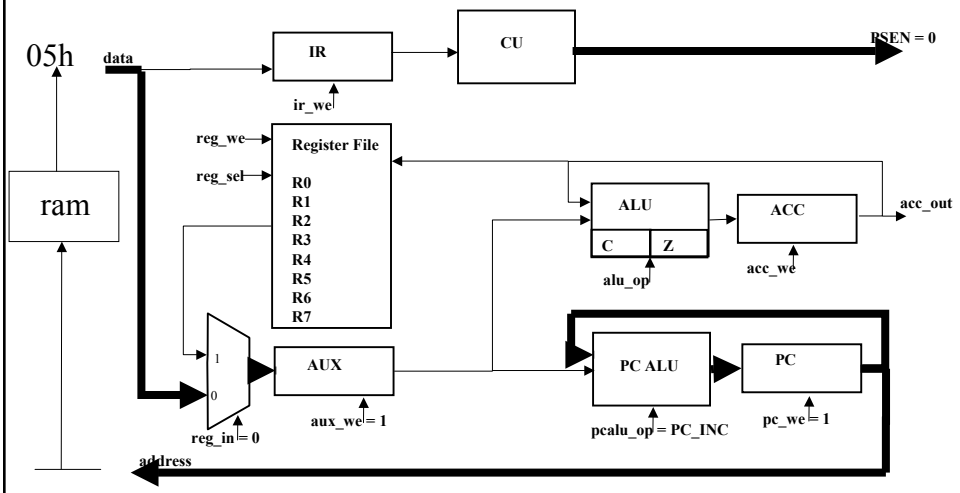


Wimp51 Instructions

MOV A, #D	01110100 dddddddd	A<=D
ADDC A, #D	00110100 dddddddd	C,A<=A+D+C
MOV Rn, A	11111nnn	Rn<=A
MOV A, Rn	11101nnn	A<=Rn
ADDC A, Rn	00111nnn	C,A<=A+Rn+C
ORL A, Rn	01001nnn	A<=AYRn
ANL A, Rn	01011nnn	A<=AIRn
XRL A, Rn	01101nnn	A<=A⊕Rn
SWAP A	11000100	$A<=A_{(3-0)}\&A_{(7-4)}$
CLR C	11000011	C<=0
SETB C	11010011	C<=1
SJMP rel	10000000 aaaaaaaa	PC<=PC+rel+2
JZ rel	01100000 aaaaaaaa	PC<=PC+rel+2 if Z

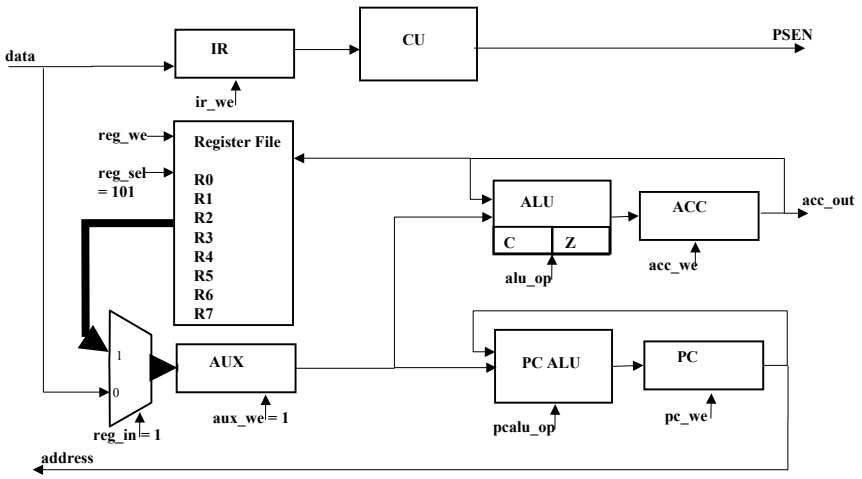
Decode cycle: immediate source

MOV A, #05h



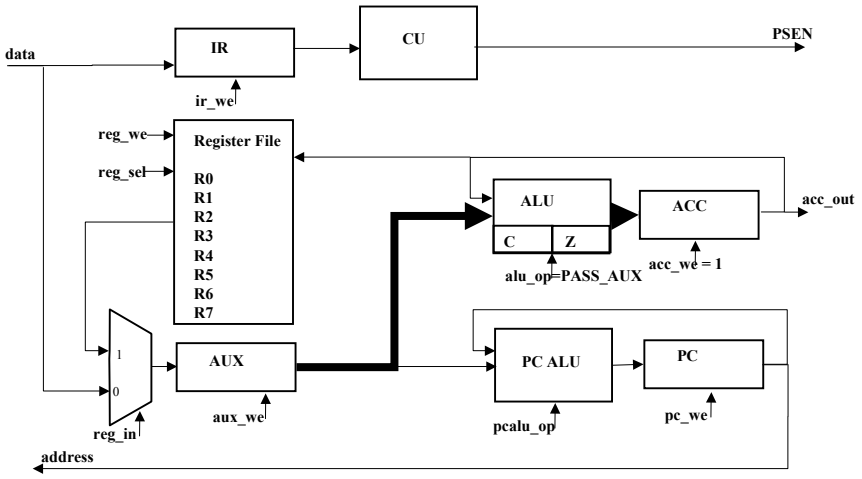
Decode cycle: register source

MOV A, R5



Execute cycle:

MOV A, R5



Execute cycle:

SJMP loop