

Microcomputers I – CE 320

Mohammad Ghamari, Ph.D.

Electrical and Computer Engineering

Kettering University

Announcements

- Your Second quiz is next Friday, Feb 12!
 - Topic:
 - Lecture 03: Introduction to HCS12
 - Lecture 04: Addressing Modes
 - Lecture 05: Unconditional Branches
 - Lecture 5.1: HCS12 Instructions
 - Lecture 06: Conditional Branches
 - Lecture 07: Comparison Branches
 - Homework Exercise 2

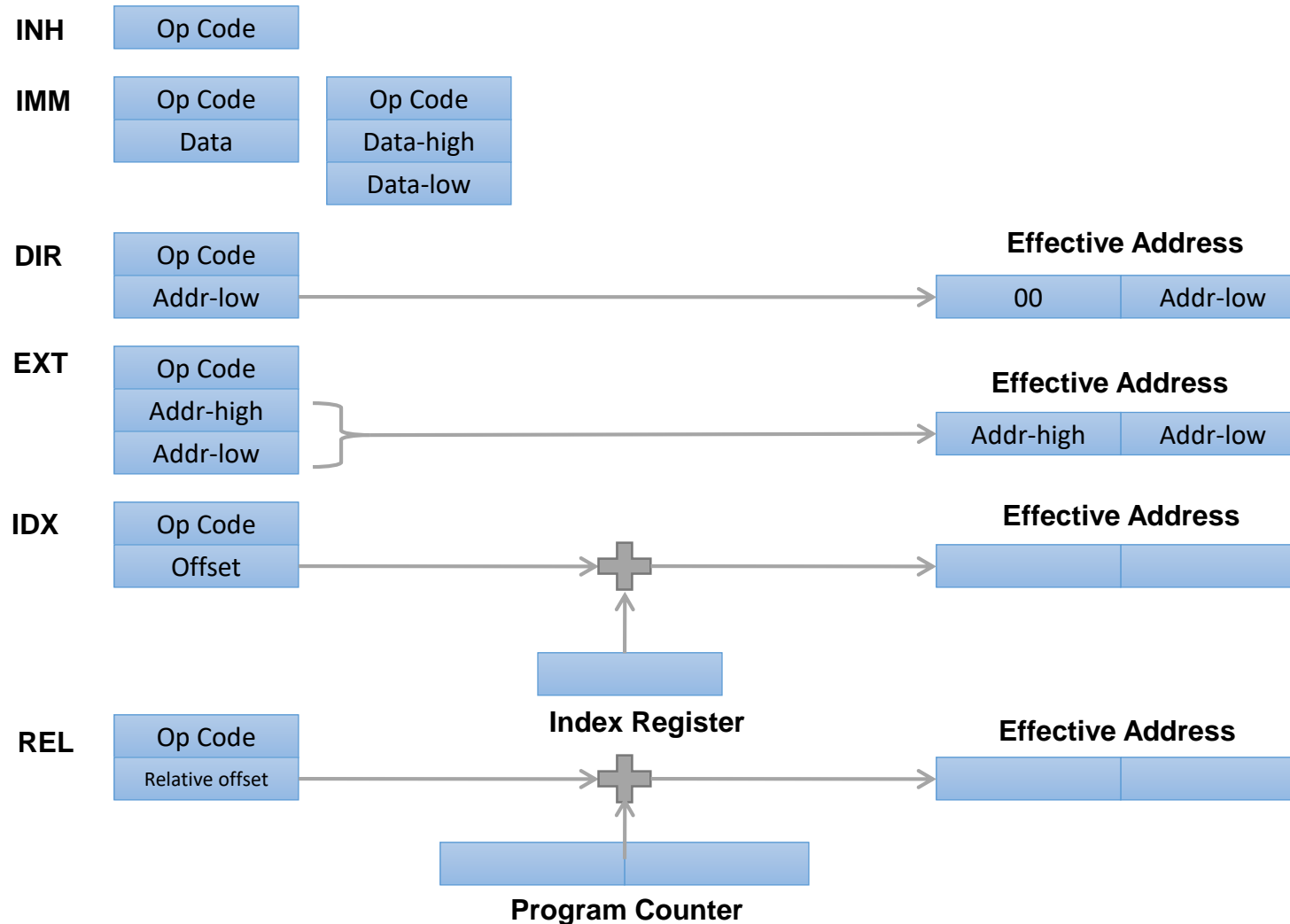
Lecture 7: Comparison Branches

Today's Goals

- Review the addressing modes
- Learn more about the basic instructions
- Use the Unsigned and Signed Comparison Branches to control the flow of programs

Addressing Mode Summary

How to Get an Effective Address





Basic Instructions

Load and store instruction

- 8 Bit accumulator **load**
 - **LDAA**: load a value from the specified memory to accumulator A
 - **LDAB**: load a value from the specified memory to accumulator B
- 8 bit accumulator **store**
 - **STAA**: store a value in accumulator A into the specified memory
 - **STAB**: store a value in accumulator B into the specified memory
- 16 bit register load and store
 - **LDD, LDX, LDY, LDS**
 - **STD, STX, STY, STS**
- Examples:
 - Tell the difference between
 - **LDAA #\$10** and **LDAA \$10**
 - **LDD \$1000** and **LDD #\$1000**

7	A	0	7	B	0
15	D			0	
15	X			0	
15	Y			0	
15	SP			0	
15	PC			0	
S X H I N Z V C					

Basic Instructions

Exchange, Move, and Clear

- **Exchange** instructions

- EXG: exchange register contents
 - EXG X Y
 - EXG A B
 - EXG X B
 - EXG B X
- XGDY: exchange register D and X
- XGDY: exchange register D and Y

- **Move**

- MOVB: move a byte from a memory to another memory
 - MOVB \$1000, \$2000
- MOVW: move a word (2 bytes) from a memory to another memory

- **Clear**

- CLR: clear a byte in the specified memory
 - CLR \$0800 ; set the content at \$0800 to 0
- CLRA (clear accumulator A)
- CLRB (clear accumulator B)

Compare Move instructions
with Store ones.

Move: Memory to Memory
Store: Register to Memory





Basic Instructions

Register to register transfer

- Copy a value from one register to another
 - TFR: Transfer a content of one register to another
 - TFR A B
 - TAB: (A) \rightarrow (B)
 - TBA: (B) \rightarrow (A)
 - SEX: Sign EXtended transfer from 8 bit register to 16 bit register
 - SEX A D
 - TPA: (CCR) \rightarrow (A)
 - TAP: (A) \rightarrow (CCR)
 - TSX: (SP) \rightarrow (X)
 - TXS: (X) \rightarrow (SP)
 - TSY: (SP) \rightarrow (Y)
 - TYS: (Y) \rightarrow (SP)

7	A	0	7	B	0
15	D				0
15	X				0
15	Y				0
15	SP				0
15	PC				0
S X H I N Z V C					

Basic Instructions

Increments, Decrements, and Negate

- Increments
 - INC: $(M) + 1 \rightarrow M$
 - INCA: $(A) + 1 \rightarrow A$
 - INCB
 - INS
 - INX
 - INY
- Decrements
 - DEC
 - DECA
 - DECB
 - DES
 - DEX
 - DEY
- Negate
 - NEG: negate a memory byte
 - NEGA: negate accumulator A
 - NEGB: negate accumulator B

Basic Instructions

Comparison

- Comparison instructions
 - Actually, they are subtractions.
 - Discard the answer
 - No change in the registers and the memories
 - CCR bits are affected instead.
- **CBA**: Compare B to A:
 - Subtract the B accumulator from the A accumulator
 - $(A) - (B)$
- **CMPA**, **CMPB**: Compare accumulator to memory :
 - Subtract the content of a memory from the accumulator
 - $(A) - (M)$, $(B) - (M)$

Comparison is nothing but subtraction discarding the answer.



The order is important!

Need to know which one is minuend or which subtrahend to interpret CCR bits.

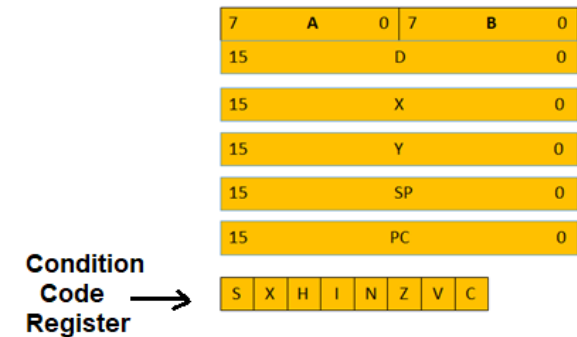




Comparison Instruction

Example

- Let register A have 10h, register B have 15h
 - (A) = 10h, (B) = 15h
- CBA**
 - (A) – (B) = FBh
 - Instead of saving the result, the result \$FB affects CCR bits.
 - N: **1**, Z: **0**, V: **1**, C: **1**



\$FB = 1111 **1011**

- CMPA, CMPB**
 - Assume FFh at address \$1000
 - CMPA \$1000
 - (A) – (\$1000) = 10h – FFh = 11h
 - N: **0**, Z: **0**, V: **0**, C: **1**

11h = 0001 **0001**

Source Form	Operation	S X H I	N Z V C
CBA	(A) – (B) Compare 8-Bit Accumulators	----	Δ Δ Δ Δ

- Therefore,
 - CBA does not mean that I want to *compare* B and A.
 - Rather, CBA means that I want to know what happens in CCR bits after (A) – (B) operation.

Comparison Branches



But, actually comparison branches only check the CCR bits.

Therefore, any instruction that can affect CCR bits must be placed before using comparison branches.

- Comparison branches are based on comparing two numbers.
- **Comparison branches** only examine several CCR bits, and for them to function correctly, the CCR must be set by a subtraction instruction.
- The format for a subtraction instruction when used to prepare for a branch is as follows:

Number of Interest – Reference Value = Result

- This format makes it a little easier for a human to determine whether the branch will be taken by mentally comparing the numbers instead of performing the subtraction as the microcomputer.
- For example, a “branch if higher” instruction would affect the PC if the Number of Interest was higher than the Reference Value.
- Subtraction instructions fall into three general categories:
 - Actual subtraction
 - Perform operation and keep the result.
 - **Comparison***
 - Perform subtraction and discard the answer.
 - Test
 - Perform subtraction using 00 as the inherent reference.

Logically, comparison instructions are needed before we use comparison branches.



Comparison Branches

Instructions

- Two sets of comparison branches:
 - Unsigned values:
 - Higher, Higher or Same, Lower, Lower or Same
 - Signed values:
 - Greater Than, Greater or Equal, Less Than, Less or Equal

Comparison	Unsigned	Signed
>	BHI – if higher	BGT – if greater
≥	BHS – if higher or same	BGE – if greater or equal
<	BLO – if lower	BLT – if less than
≤	BLS – if lower or same	BLE – less than or equal
=	BEQ – if equal	BEQ – if equal
≠	BNE – if not equal	BNE – if not equal

BHI, BHS, BLO, BLS, LBHI, LBHS, LBLO, LBLS – Branch to an instruction based on a **comparison of unsigned values**.

BEQ, BNE - Can be used for either signed or unsigned values.

BGT, BGE, BLT, BLE, LBGT, LBGE, LBLT, LBLE – Branch to an instruction based on a **comparison of signed values**.

Comparison Branches

Example Program

- Trace the program below. Assume the memory locations \$2000, \$2001, and \$2002 are already set to \$40, \$F0, and \$55 respectively.

1:	1500	CE 2000	LDX #\$2000
2:	1503	180B FF 1000	MOVB #\$FF,\$1000
3:	1508	C6 02	LDAB #2
4:	150A	27 0E	BEQ 14
5:	150C	A6 00	LDAA 0,X
6:	150E	B1 1000	CMPA \$1000
7:	1511	24 03	BHS 3
8:	1513	7A 1000	STAA \$1000
9:	1516	08	INX
10:	1517	53	DECB
11:	1518	20 F0	BRA -16
12:	151A	3F	SWI

...	
2000	40
2001	F0
2002	55
...	

1:1500CE 2000LDX #\$2000

2:1503180B FF 1000MOVB #\$FF,\$1000

3:1508C6 02LDAB #2

4:150A27 0EBEQ 14

5:150CA6 00LDAA 0,X

6:150EB1 1000CMPA \$1000

7:151124 03BHS 3

8:15137A 1000STAA \$1000

9:151608INX

10:151753DECB

11:151820 F0BRA -16

12:151A3FSWI

...	
1000	
...	
2000	40
2001	F0
2002	55
...	

Trace	Line	PC	A	B	X	N	Z	V	C
1	1	1503	-	-	2000	0	0	0	-
2	2	1508	-	-	2000	0	0	0	-
3	3	150A	-	02	2000	0	0	0	-
4	4	150C	-	02	2000	0	0	0	-
5	5	150E	40	02	2000	0	0	0	-
6	6	1511	40	02	2000	0	0	0	1
7	7	1513	40	02	2000	0	0	0	1
8	8	1516	40	02	2000	0	0	0	1
9	9	1517	40	02	2001	0	0	0	1
10	10	1518	40	01	2001	0	0	0	1

1: 1500 CE 2000 LDX #\$2000

2: 1503 180B FF 1000 MOVB #\$FF,\$1000

3: 1508 C6 02 LDAB #2

4: 150A 27 0E BEQ 14

5: 150C A6 00 LDAA 0,X

6: 150E B1 1000 CMPA \$1000

7: 1511 24 03 BHS 3

8: 1513 7A 1000 STAA \$1000

9: 1516 08 INX

10: 1517 53 DECB

11: 1518 20 F0 BRA -16

12: 151A 3F SWI

...	
1000	
...	
2000	40
2001	F0
2002	55
...	

FF → 40

Trace	Line	PC	A	B	X	N	Z	V	C
11	11	150A	40	01	2001	0	0	0	1
12	4	150C	40	01	2001	0	0	0	1
13	5	150E	F0	01	2001	1	0	0	1
14	6	1511	F0	01	2001	1	0	0	0
15	7	1516	F0	01	2001	1	0	0	0
16	9	1517	F0	01	2002	1	0	0	0
17	10	1518	F0	00	2002	0	1	0	0
18	11	150A	F0	00	2002	0	1	0	0
19	4	151A	F0	00	2002	0	1	0	0
20	12	-	-	-	-	-	-	-	-

Homework: Questions

- What does this program do?
 - Get a minimum value from the values from \$2000 to (\$2000 + the initial content in register B)
- What changes are needed to process 200 bytes?
 - Line 3: LDAB #2 → LDAB #200 (or #\$C8 or #C8h)
- What changes are needed to process signed numbers?
 - Line 7: BHS → BGT
 - Line 2: #\$FF → #\$7F (or #7Fh)
- What changes are needed if the list of data begins at \$3000?
 - Line 1: #\$2000 → #\$3000h (or #3000h)
- What changes are needed if the answer must be stored to location \$3FFF?
 - Line 2, 6, and 8: \$1000 → \$3FFF (or 3FFFh)

Questions?

Wrap-up

What we've learned

- Comparison branches
 - Unsigned
 - BHI, BGT, BHS, BGE
 - Signed
 - BLO, BLT, BLS, BLE
 - Either signed or unsigned
 - BEQ, BNE

What to Come

- Assembly language
- Flowchart