

INTRODUCTION to DIGITAL COMPUTERS

LABORATORY NO. 7

INTRODUCTION to the ICS05J1A SIMULATOR

OBJECTIVES:

1. To learn the M68HC05 instruction subset.
2. To learn how to assemble M68HC05 programs.
3. To learn to run programs, set breakpoints and trace through programs.

EQUIPMENT :

Gateway2000, RAPID Microcontroller Development System.

REFERENCES:

1. M68705J1A In-Circuit Simulator. User's Manual (copy in lab)
2. MC68HC705J1A Technical Data (You should have)

SIMPLE M68HC05 PROGRAM EXAMPLES:

Example 1.

```

    org rom      ;sets the starting address $0300 for the code that follows
start: lda #$10  ;loads the hex number 10 (decimal 16) intoACCA
    ldx #$11     ;loads the hex number 11 (decimal 17) into the index register X
    mul         ;multiplies the contents ofACCA by the contents of X.
               ;After the operation,X contains the upper 8 bits of the
               ;16-bit result andACCA contains the lower 8 bits

    stop
    org $7FE    ;sets the address $07FE for the object code that follows
back1: fdb start ;forms double byte address constantstart that defines
               ;where the program counter begins on reset

```

After this program executes, X and ACCA together contain the hexadecimal number 110 (decimal 272).

Example 2.

```

temp1 equ $C0   ;one byte temp storage location
    org rom     ;program will start at $0300
start: lda #$D  ;loads the hexadecimal numberD (decimal 13) into
               ;the accumulatorACCA
    sta temp1   ;stores the contents ofACCA (hex D (decimal 13)) in
               ;temp1 (the address $C0 )
    lda #$20    ;loads the hexadecimal number 20 (decimal 32) intoACCA
    sub temp1   ;subtracts the contents oftemp1 from ACCA and places
               ;the result inACCA

    stop
    org $7FE    ;sets the address $07FE for the object code that follows
back1: fdb start ;forms double byte address constantstart that defines
               ;where the program counter begins on reset

```

After executing this program, ACCA contains the hexadecimal number 13 (decimal 19).

Example 3.

```

temp1 equ $C0      ;one byte temp storage location
temp2 equ $C1      ;one byte temp storage location
        org rom    ;program will start at $0300
start:  lda #$5     ;loads the hexadecimal number 5 (decimal 5) intoACCA
        sta temp1  ;stores the contents ofACCA (the hexadecimal number 5
                    ;(decimal 5)) intemp1 (the address $C0)
        lda #$2     ;loads the hexadecimal number 2 (decimal 2) intoACCA
        sta temp2  ;stores the contents ofACCA (the hexadecimal number 2
                    ;(decimal 2)) intemp2 (the address $C1 )
        lda #$1     ;loads the hex number 1 (decimal 1) intoACCA
loop1:  ldx temp2   ;loads the hex number 2 (decimal 2) into index register X
        mul        ;multiplies the contents ofACCA by the contents of X.
                    ;After the operation, X contains the upper 8 bits of the
                    ;16-bit result andACCA contains the lower 8 bits
        dec temp1  ;subtracts one from the contents oftemp1
        bne loop1  ;the branch (jump to the labelloop1) will occur if the
                    ;contents oftemp1 is not equal to 0
        stop      ;stop oscillator of themicrocontroller
        org $7FE  ;sets the address $07FE for the object code that follows
back1:  fdb start  ;forms double byte address constantstart

```

After executing this example ACCA and X contain the hexadecimal number 80 (decimal 128).

PRELAB:

Using the above examples as a guide, develop the following programs:

Program 1.

Multiply \$4 by \$5 and put the result at address $\$00C1$ (see Example 1 and Example 2).

Program 2.

Calculate the following value: $2^6 + 3$ using a loop (see Example 3).

Be sure to add comments to your programs.

PROCEDURE:

A. How to prepare and run a program.

Use the program given in Example 2.

1. Click two times on *Shortcut to Rapid.exe*.
2. Press \uparrow key so that

..\

is highlighted.

3. Press *Esc*. You see the line

Pathname of file:C:\ICS05J1A*.ASM

type in instead of * the name of the ASM file (for this example type: first1).

4. Press *Enter*. Now you get the Rapid editor display. You can type in any M68HC05 program. Type in the program from Example 2. Remember: all labels and names of variables (for example start and temp1, see Example 2) must be typed starting on the first column!!!!
5. Press *F4* and *Enter* to assemble the program.
6. If you see the line

Successful assembly - No errors.

go to the next step. If you get errors you have to correct them.

7. Press *F6*. You see the line

Debugger command: C:\ICS05J1A\FIRST1.S19

Press *Enter*. Now you see ICS05J1A Main Screen (Fig. 1).

8. Type *g* after prompt *>* in Debug Window to start execution of code in the simulator (Debug F10, Fig. 1). Press *Enter*. If you do not get errors you can see the final result in ACCA, CPU Window (CPU, Acc Fig. 1).

B. How to set break points.

Setting break points forces the execution of the program to stop and control to return to the monitor program. You can check the intermediate results of an operation.

Example 4.

Suppose that after you run the program of Example 2 you would like to check the result of the program after the instruction *lda #\$20* is executed. You can do it using the following steps:

1. Type \uparrow after prompt *>* in Debug Window (Debug F10, Fig. 1) to restore the initial value of PC. Press *Enter*. See the CPU Window to verify the result (CPU, Fig. 1).
2. Type ~~pc 300~~ after prompt *>* in Debug Window (Debug F10, Fig. 1). Press *Enter* (the address \$306 corresponds to the first byte of the instruction

sub temp1, the next instruction after *lda #\$20*).

3. Type *g* after prompt *>* in Debug Window (Debug F10, Fig. 1) to start execution of code in the simulator. Press *Enter*. If you do not get errors, you can see the intermediate result in ACCA, CPU Window (CPU, Acc Fig. 1).

C. How to trace through a program.

Instead of using break points, you can trace a program, executing one instruction at a time, using the command *t*. Type *t* after prompt *>* in Debug Window (Debug F10, Fig. 1) and press *Enter*. The command ~~*br 306*~~ means "execute *n* assembly instructions".

D. Run the program from Example 3.

Enter and run the program from Example 3, using the command *g* and check the final result. Trace this program and record the intermediate results after the instruction *mul*. Repeat these results using the command ~~*br 30C n*~~ ($n=1,2,\dots,7$).

E. Run your prelab programs.

Enter and run your prelab programs, using the command *g* and check the final result. For program 3, use Trace these programs and record the intermediate results. Repeat these tests using the command *br* with corresponding address and the value of *n*.

F. **Program 3.** Write a program to calculate the equation $F = X * \$5 + Y * \9 , where X and Y are stored in locations \$C0 and \$C1 and F is stored in location \$C2. Test this program for the following values of X and Y.

X	Y	F
0	\$0F	
\$0C	\$00	
\$12	\$02	
\$05	\$0A	

G. Print your programs.

Open the "notepad" application and open the .asm file that you created in

C:\ICS05J1A\

Use the print command in the notepad application to print the file.

H. Save your programs.

Save your programs on your disk or email or ftp them to your idol account (ASCII mode).

THE REPORT SHOULD INCLUDE:

1. All your programs and computer results with explanations.
2. Explain why the start address is stored at \$07FE.
3. Explain what "lda #\$10" (Example 1) does? Is this instruction stored in ROM or in RAM?
4. Explain what "temp1" (Example 2) does? Is this value stored in ROM or in RAM?
5. Determine and record the addresses of all instructions in program of Example 1.
6. Find and record the assembled version of Example 1.

CPU			
Acc Xreg	SP	PC	
XX XX	00FF	0300	
CCR		CYCLES	
111.1..	00000000		

SOURCE: FIRST1.ASM

VARIABLES F8

MEMORY F3

DEBUG F10

>

Fig 1 ICS05J1A Main Screen

Cherrice Traver

Fall 2004