İ.T.Ü.
Faculty of Computer and Informatics
Computer Engineering

MICROCOMPUTER LAB REPORT

Lab No	**:** 02
Lab Date	**:** 03.10.2013
Group	**:** B9
Group Members	**:** 040100014 Teoman TURAN 040100018 Mustafa DURMUŞ 040100117 Tuğrul YATAĞAN 040100124 Emre GÖKREM

Research Assistant : Salih Serdar GÜÇLÜ

1. THE AIM/CONTENT of THE EXPERIMENT

The purpose of doing this experiment is to examine addressing modes (methods) of MC6802 within a program written in Motorola 6800 Assembly programming language: immediate, direct, inherent, extended, relative.

2. EQUIPMENT

Only ITU-Training Kit has been used in the experiment. This kit consists of the following hardware components:

- CPU: MC6802
- Memory: 24K*8 R/W + 16K*8 Read Only
- Address decoder
- Control unit
- Display and Keypad
- Parallel Port
- Serial Port
- Programmable counter

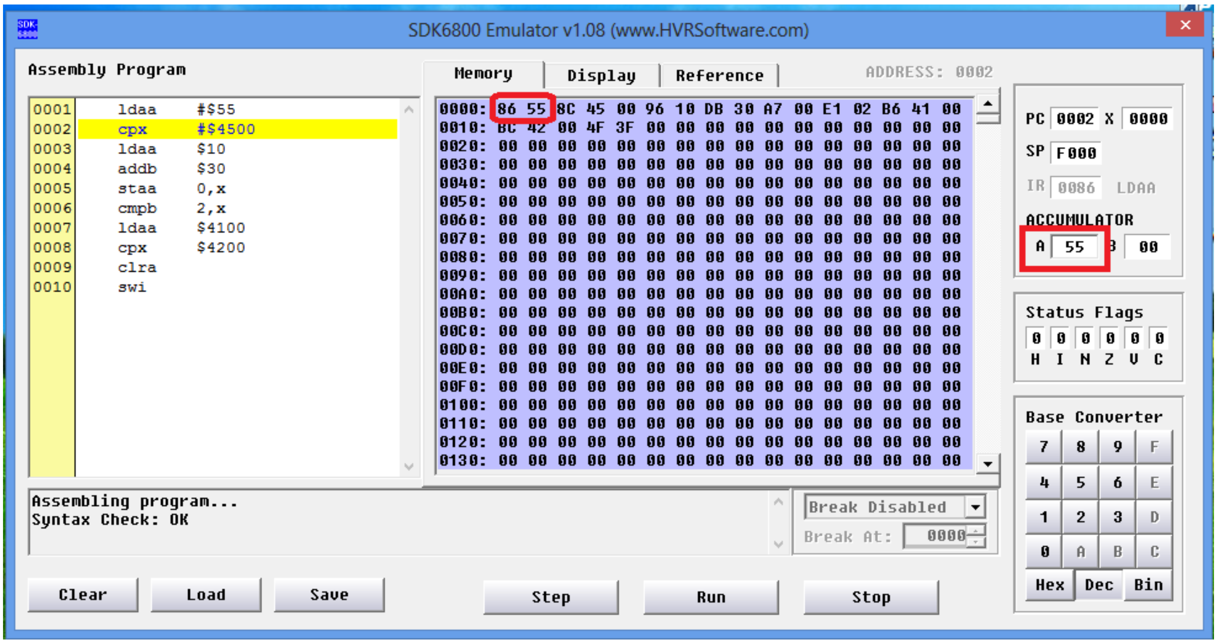
In addition to this kit, for observing the simulation results during the experiment, one of our group members has used his own computer to run the simulation software of the microprocessor.

3. THE PROGRAMS for THE EXPERIMENT

3.1. SECTION 2: ADDRESSING MODES

The machine codes of the instructions in the code fragment below and their addressing modes had been determined before coming to the lab. Also, that code fragment had been written and run on the simulator software of the microprocessor. In the lab, the code has been run on ITU-Training Kit. Here are the machine codes of the instructions, their addressing modes, what they do and screenshots for each of them from the simulation software of the microprocessor:

```
LDAA #$55      86 55           Immediate
        // It immediately loads direct data $55 to Accumulator A.
```



Assembly Program

```
      ldaa    #$55
      cpx     #$4500
      ldaa    $10
      addb    $30
      staa    0,x
      cmpb    2,x
      ldaa    $4100
      cpx     $4200
      clra
      swi
```

ADDRESS: 0002

```
0000: 86 55 8C 45 00 96 10 DB 30 A7 00 E1 02 B6 41 00
0010: BC 42 00 4F 3F 00 00 00 00 00 00 00 00 00 00 00
0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

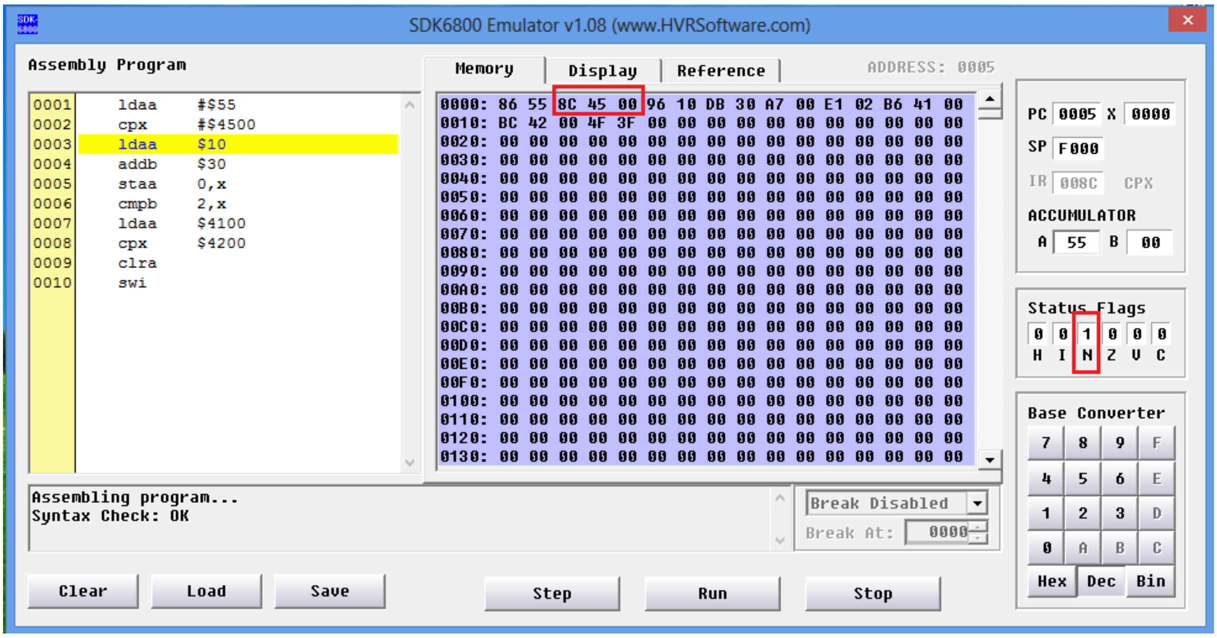
```
PC 0002  X 0000
SP F000
IR 0086  LDAA
ACCUMULATOR
A 55  B 00
Status Flags
0 0 0 0 0 0
H I N Z V C
```

```
Assembling program...
Syntax Check: OK
```

Break Disabled
Break At: 0000

CPX #$4500   8C 45 00   Immediate

// It compares the value in the memory address kept in Index Register with another direct data, $4500.



Assembly Program

```
      ldaa    #$55
      cpx     #$4500
      ldaa    $10
      addb    $30
      staa    0,x
      cmpb    2,x
      ldaa    $4100
      cpx     $4200
      clra
      swi
```

ADDRESS: 0005

```
0000: 86 55 8C 45 00 96 10 DB 30 A7 00 E1 02 B6 41 00
0010: BC 42 00 4F 3F 00 00 00 00 00 00 00 00 00 00 00
0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

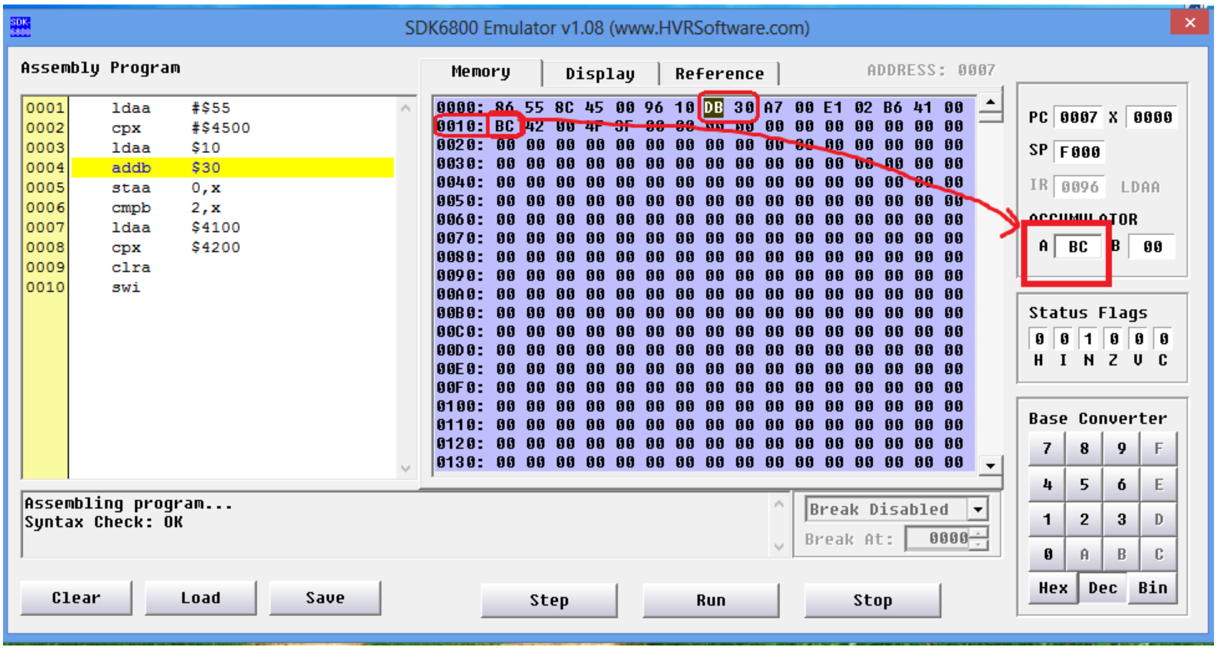
```
PC 0005  X 0000
SP F000
IR 008C  CPX
ACCUMULATOR
A 55  B 00
Status Flags
0 0 1 0 0 0
H I N Z V C
```

```
Assembling program...
Syntax Check: OK
```

Break Disabled
Break At: 0000

LDAA $10   96 10   Direct

// It loads the value in memory address $10 to the Accumulator A.



```
        ldaa    #$55
        cpx     #$4500
        ldaa    $10
        addb    $30
        staa    0,x
        cmpb    2,x
        ldaa    $4100
        cpx     $4200
        clra
        swi
```

```
0000: 86 55 8C 45 00 96 10 DB 30 A7 00 E1 02 B6 41 00
0010: BC 42 00 4F 3F 00 00 00 00 00 00 00 00 00 00 00
0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

PC 0007 X 0000
SP F000
IR 0096 LDAA
ACCUMULATOR
A BC B 00

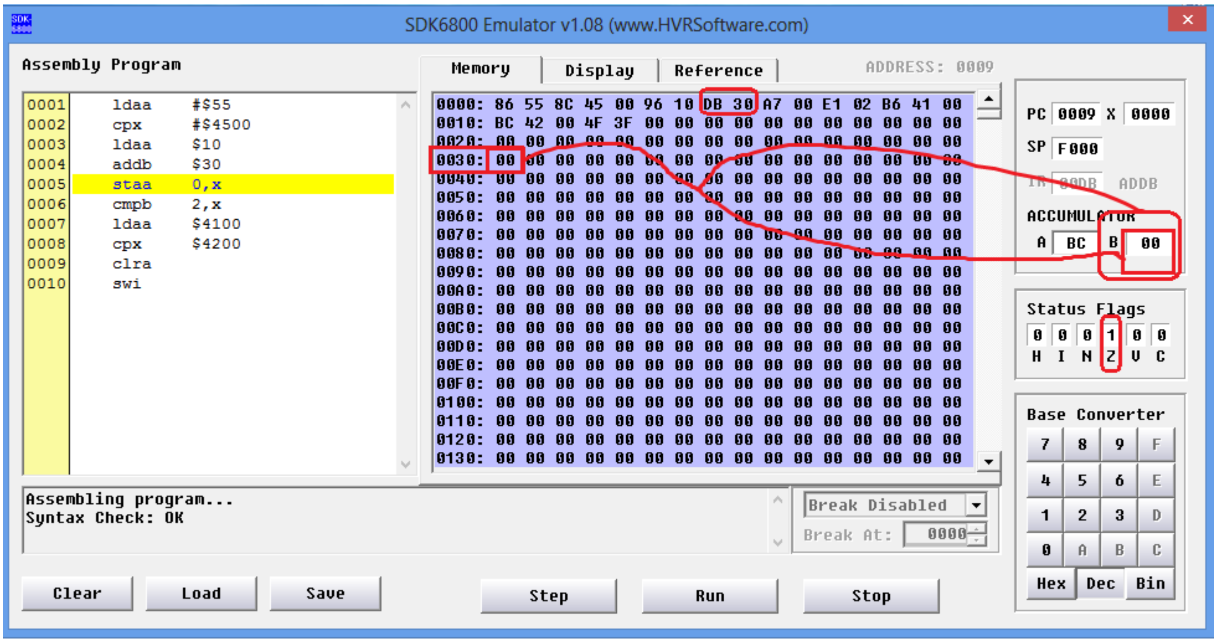
Status Flags

H	I	N	Z	V	C
0	0	1	0	0	0

```
Assembling program...
Syntax Check: OK
```

ADDB $30 DB 30 Direct

// It sums the value in memory address $30 and the one in Accumulator B, then stores the result in Accumulator B.



```
        ldaa    #$55
        cpx     #$4500
        ldaa    $10
        addb    $30
        staa    0,x
        cmpb    2,x
        ldaa    $4100
        cpx     $4200
        clra
        swi
```

```
0000: 86 55 8C 45 00 96 10 DB 30 A7 00 E1 02 B6 41 00
0010: BC 42 00 4F 3F 00 00 00 00 00 00 00 00 00 00 00
0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

PC 0009 X 0000
SP F000
IR 00DB ADDB
ACCUMULATOR
A BC B 00

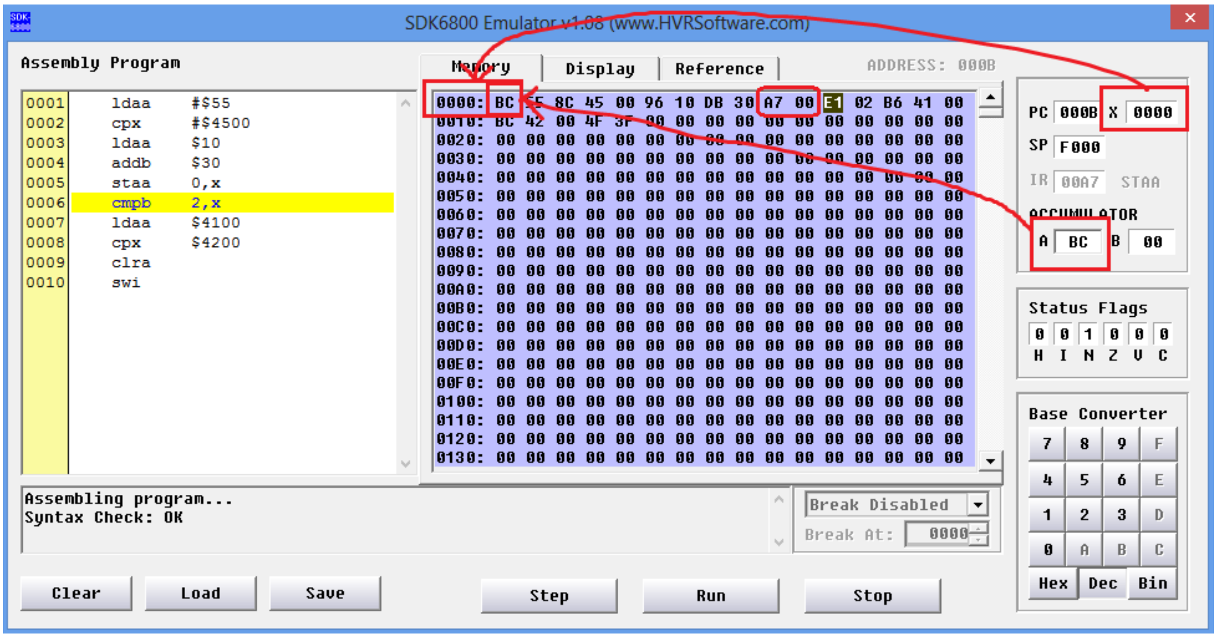
Status Flags

H	I	N	Z	V	C
0	0	0	1	0	0

```
Assembling program...
Syntax Check: OK
```

STAA 0,x A7 00 Indexed

// It stores the value in Accumulator A to the address kept in Index Register with the index of 0. So, there is no offset change.



SDK6800 Emulator v1.08 (www.HVRSoftware.com)

Assembly Program

```
        ldaa    #$55
        cpx     #$4500
        ldaa    $10
        addb    $30
        staa    0,x
        cmpb    2,x
        ldaa    $4100
        cpx     $4200
        clra
        swi
```

```
Assembling program...
Syntax Check: OK
```

ADDRESS: 000B

```
0000: BC 55 8C 45 00 96 10 DB 30 A7 00 E1 02 B6 41 00
0010: BC 42 00 4F 3F 00 00 00 00 00 00 00 00 00 00 00
0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

PC 000B X 0000

SP F000

IR 00A7 STAA

ACCUMULATOR

A BC B 00

Status Flags

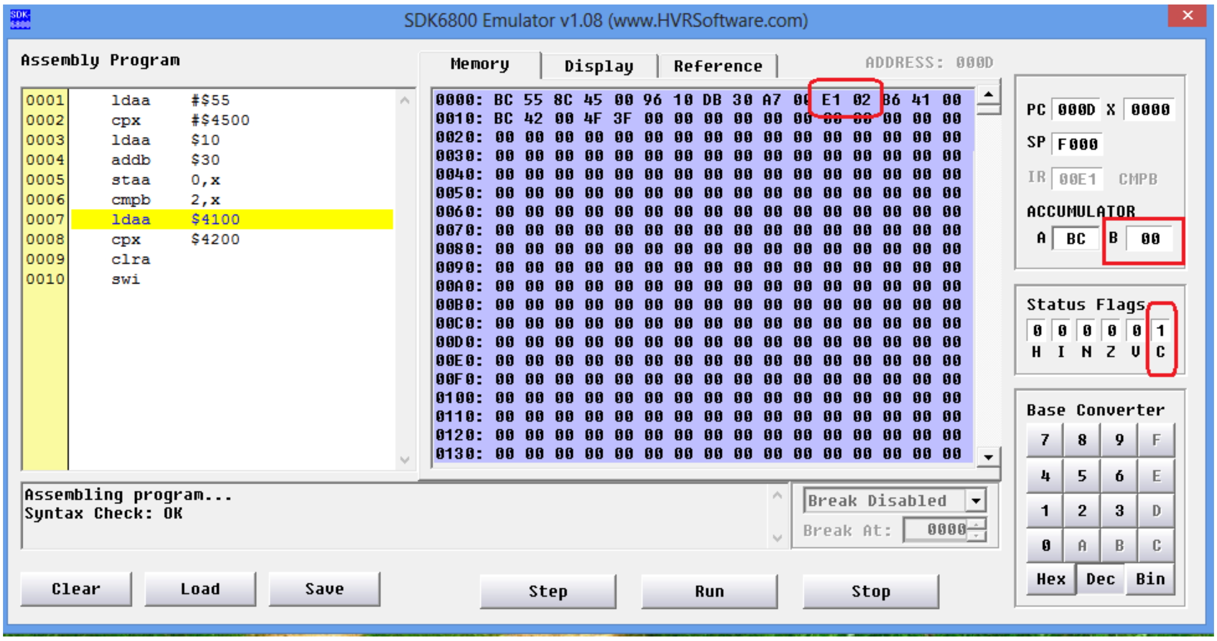
H	I	N	Z	V	C
0	0	1	0	0	0

Break Disabled

Break At: 0000

CMPB 2,x   E1 02   Indexed

// It compares the value in Accumulator B with the value in the address with the index of 2 with respect to the one kept in Index Register.



SDK6800 Emulator v1.08 (www.HVRSoftware.com)

Assembly Program

```
        ldaa    #$55
        cpx     #$4500
        ldaa    $10
        addb    $30
        staa    0,x
        cmpb    2,x
        ldaa    $4100
        cpx     $4200
        clra
        swi
```

```
Assembling program...
Syntax Check: OK
```

ADDRESS: 000D

```
0000: BC 55 8C 45 00 96 10 DB 30 A7 00 E1 02 B6 41 00
0010: BC 42 00 4F 3F 00 00 00 00 00 00 00 00 00 00 00
0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

PC 000D X 0000

SP F000

IR 00E1 CMPB

ACCUMULATOR

A BC B 00

Status Flags

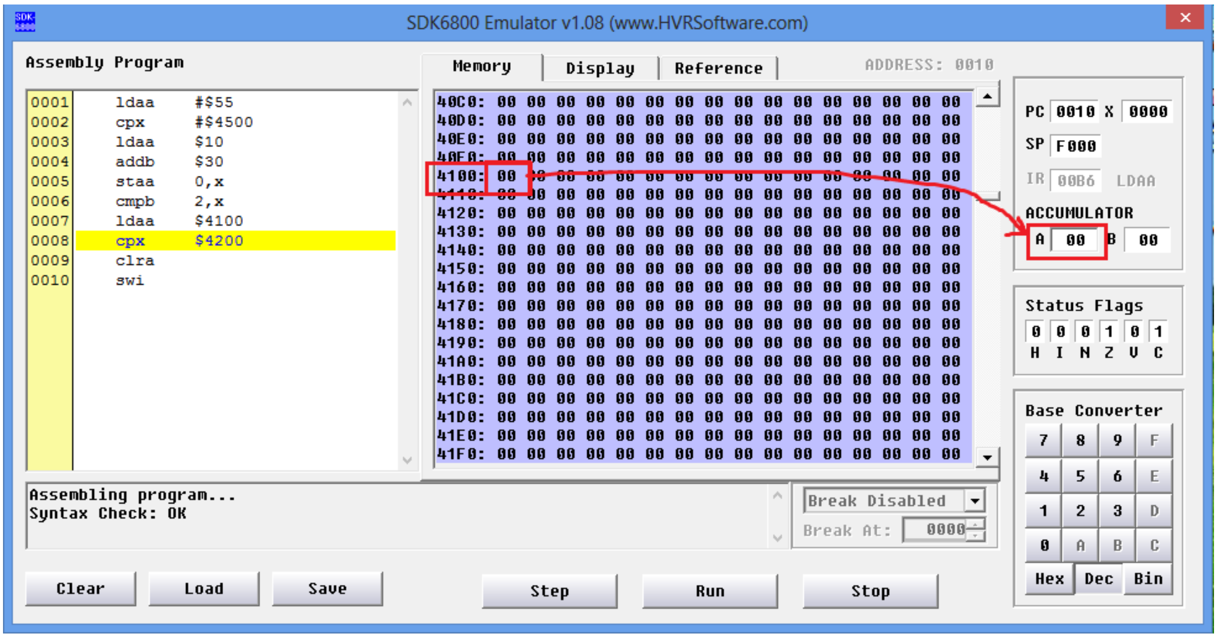
H	I	N	Z	V	C
0	0	0	0	0	1

Break Disabled

Break At: 0000

LDAA $4100   B6 41 00   Extended

// It loads the value in memory address $4100 to Accumulator A.



ADDRESS: 0010

```
        ldaa    #$55
        cpx     #$4500
        ldaa    $10
        addb    $30
        staa    0,x
        cmpb    2,x
        ldaa    $4100
        cpx     $4200
        clra
        swi
```

```
40C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
40D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
40E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
40F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4140: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4160: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4180: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

PC 0010 X 0000
SP F000
IR 00B6 LDAA
ACCUMULATOR
A 00 B 00

Status Flags

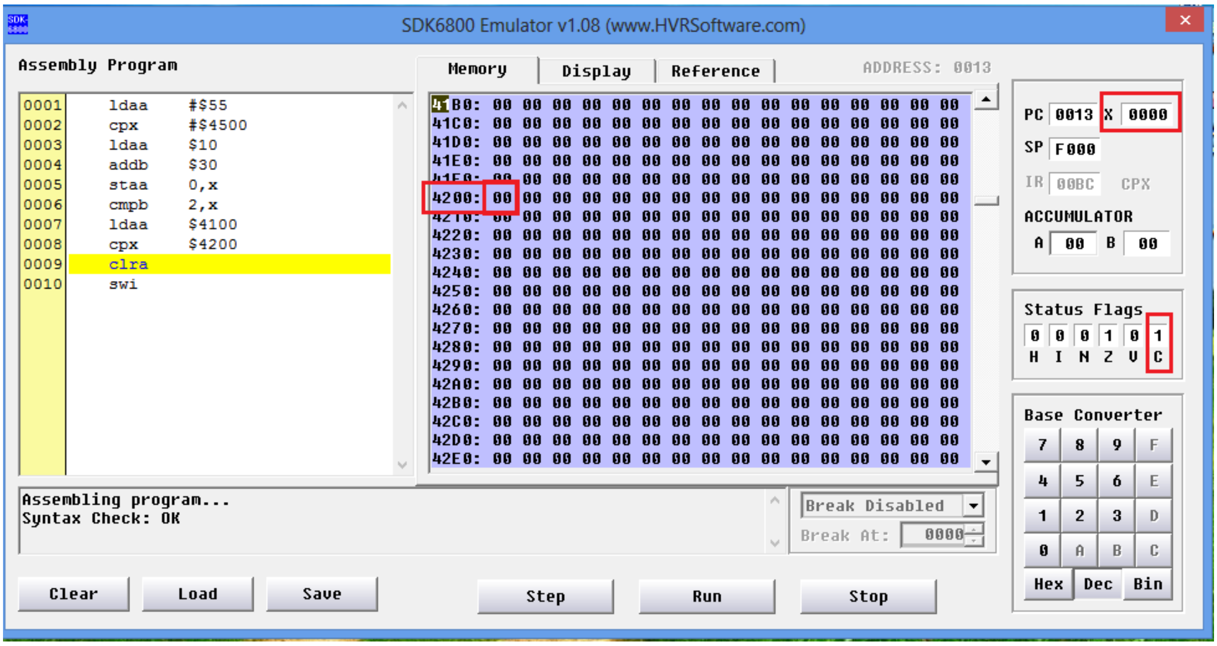
0	0	0	1	0	1
H	I	N	Z	V	C

```
Assembling program...
Syntax Check: OK
```

Break Disabled
Break At: 0000

CPX $4200   BC 42 00   Extended

// It compares the value in memory address $4200 with the value in the memory address kept in Index Register.



ADDRESS: 0013

```
        ldaa    #$55
        cpx     #$4500
        ldaa    $10
        addb    $30
        staa    0,x
        cmpb    2,x
        ldaa    $4100
        cpx     $4200
        clra
        swi
```

```
41B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4200: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4210: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4220: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4230: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4240: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4250: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4260: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4270: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4280: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4290: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

PC 0013 X 0000
SP F000
IR 00BC CPX
ACCUMULATOR
A 00 B 00

Status Flags

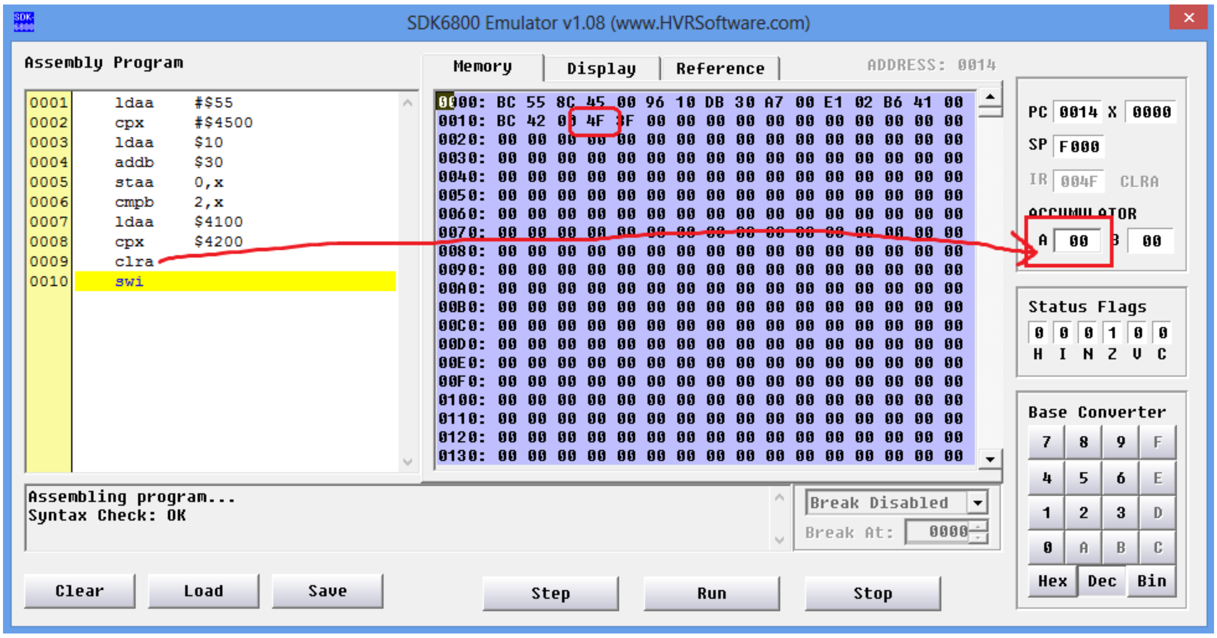
0	0	0	1	0	1
H	I	N	Z	V	C

```
Assembling program...
Syntax Check: OK
```

Break Disabled
Break At: 0000

CLRA   4F   Inherent

// This is an instruction taking no operand. It clears Accumulator A, in other words, makes the content of Accumulator A is zero.



Assembly Program

```
        ldaa    #$55
        cpx     #$4500
        ldaa    $10
        addb    $30
        staa    0,x
        cmpb    2,x
        ldaa    $4100
        cpx     $4200
        clra
        swi
```

ADDRESS: 0014

```
0000: BC 55 8C 45 00 96 10 DB 30 A7 00 E1 02 B6 41 00
0010: BC 42 00 4F 3F 00 00 00 00 00 00 00 00 00 00 00
0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

PC 0014 X 0000
SP F000
IR 004F CLRA
ACCUMULATOR
A 00 B 00

Status Flags

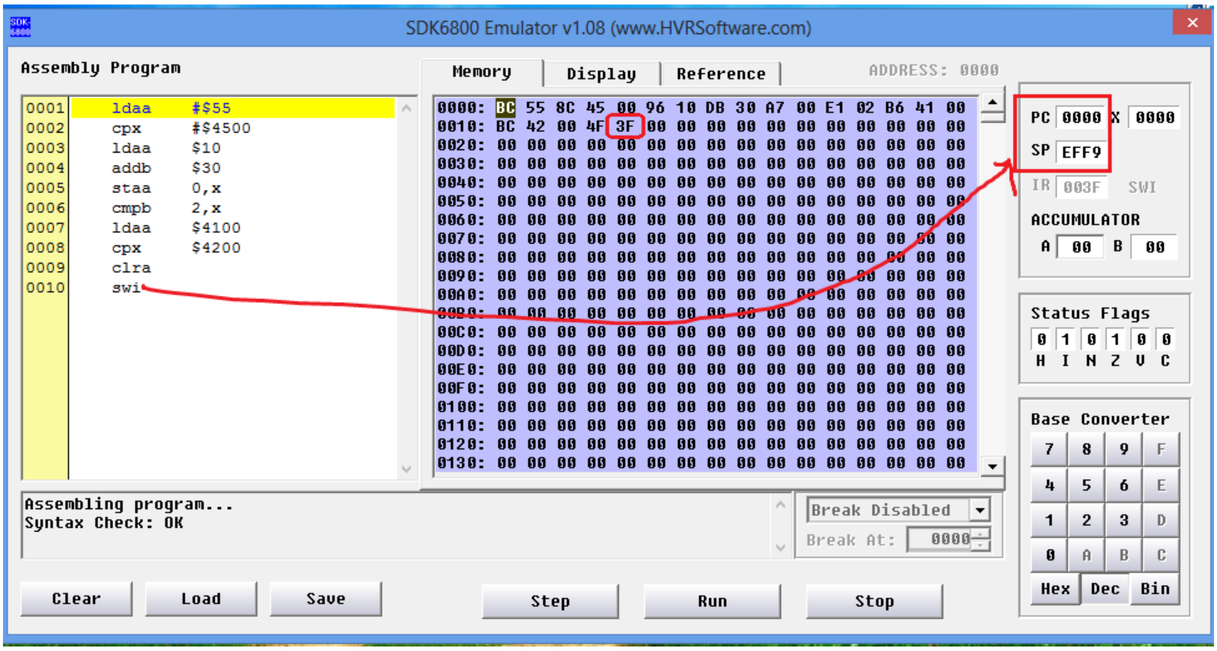
H	I	N	Z	V	C
0	0	0	1	0	0

```
Assembling program...
Syntax Check: OK
```

Break Disabled
Break At: 0000

SWI 3F Inherent

// This is an instruction taking no operand too. It terminates the program.



Assembly Program

```
        ldaa    #$55
        cpx     #$4500
        ldaa    $10
        addb    $30
        staa    0,x
        cmpb    2,x
        ldaa    $4100
        cpx     $4200
        clra
        swi
```

ADDRESS: 0000

```
0000: BC 55 8C 45 00 96 10 DB 30 A7 00 E1 02 B6 41 00
0010: BC 42 00 4F 3F 00 00 00 00 00 00 00 00 00 00 00
0020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

PC 0000 X 0000
SP EFF9
IR 003F SWI
ACCUMULATOR
A 00 B 00

Status Flags

H	I	N	Z	V	C
0	1	0	1	0	0

```
Assembling program...
Syntax Check: OK
```

Break Disabled
Break At: 0000

3.2. SECTION 3: EXPERIMENT

Here, a program which separates odd and even numbers of Array A that is the main array has been written. This program starts from the address $4000 and applies the following algorithm:

Beginning address of Array A: $4100

Beginning address of the new array for even numbers: $4200

Beginning address of the new array for odd numbers: $4300

```
j=0;
k=0;
for(i=0; i<n; i++)
{
     if(A[i] % 2 == 0)
          Even[j++] = A[i];
     else
          Odd[k++] = A[i];
}
```

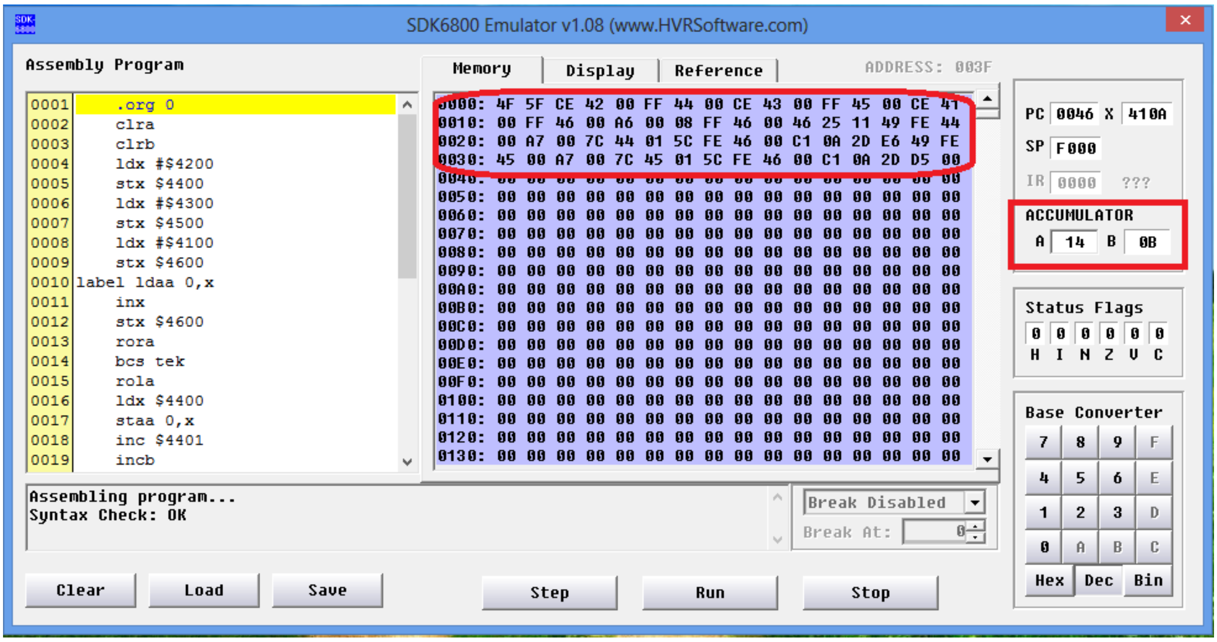
If least significant bit of a binary number is 1, it is odd. Otherwise, it is even. The program has been written by applying the algorithm for 10 numbers. It has been converted into machine codes and run on ITU-Training Kit. Here is the code of the program in Motorola 6800 Assembly programming language:

```
4F                        CLRA
5F                        CLRB
CE 42 00                  LDX #$4200
FF 44 00                  STX $4400
CE 43 00                  LDX #$4300
FF 45 00                  STX $4500
CE 41 00                  LDX #$4100
FF 46 00                  STX $4600
A6 00           LABEL     LDAA 0,x
08                        INX
FF 46 00                  STX $4600
46                        RORA
25 11                     BCS TEK
49                        ROLA
FE 44 00                  LDX $4400
```

```
A7 00                      STAA 0,x
7C 44                      INC $4401
5C                         INCB
FE 46 00                   LDX $4600
C1 0A                      CMPB #$0A
2D E6                      BLT LABEL
49              TEK        ROLA
FE 45 00                   LDX $4500
A7 00                      STAA 0,x
7C 45 01                   INC $4501
45                         INCB
FE 46 00                   LDX $4600
C1 0A                      CMPB #$0A
2D D5                      BLT LABEL
                           .ORG $4100
                           .BYTE 1,2,3,4,5,6,7,8,9,10
```

The left-most column contains the machine codes for the instructions that are contained on the right-most column. The middle column contains the label worlds “TEK” and “LABEL” pointing to the lines being branched.

Here are screenshots demonstrating the result from the simulation software of the microprocessor:



Assembly Program

```
      .org 0
      clra
      clrb
      ldx #$4200
      stx $4400
      ldx #$4300
      stx $4500
      ldx #$4100
      stx $4600
label ldaa 0,x
      inx
      stx $4600
      rora
      bcs tek
      rola
      ldx $4400
      staa 0,x
      inc $4401
      incb
```

```
Assembling program...
Syntax Check: OK
```

ADDRESS: 003F

```
0000: 4F 5F CE 42 00 FF 44 00 CE 43 00 FF 45 00 CE 41
0010: 00 FF 46 00 A6 00 08 FF 46 00 46 25 11 49 FE 44
0020: 00 A7 00 7C 44 01 5C FE 46 00 C1 0A 2D E6 49 FE
0030: 45 00 A7 00 7C 45 01 5C FE 46 00 C1 0A 2D D5 00
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0100: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

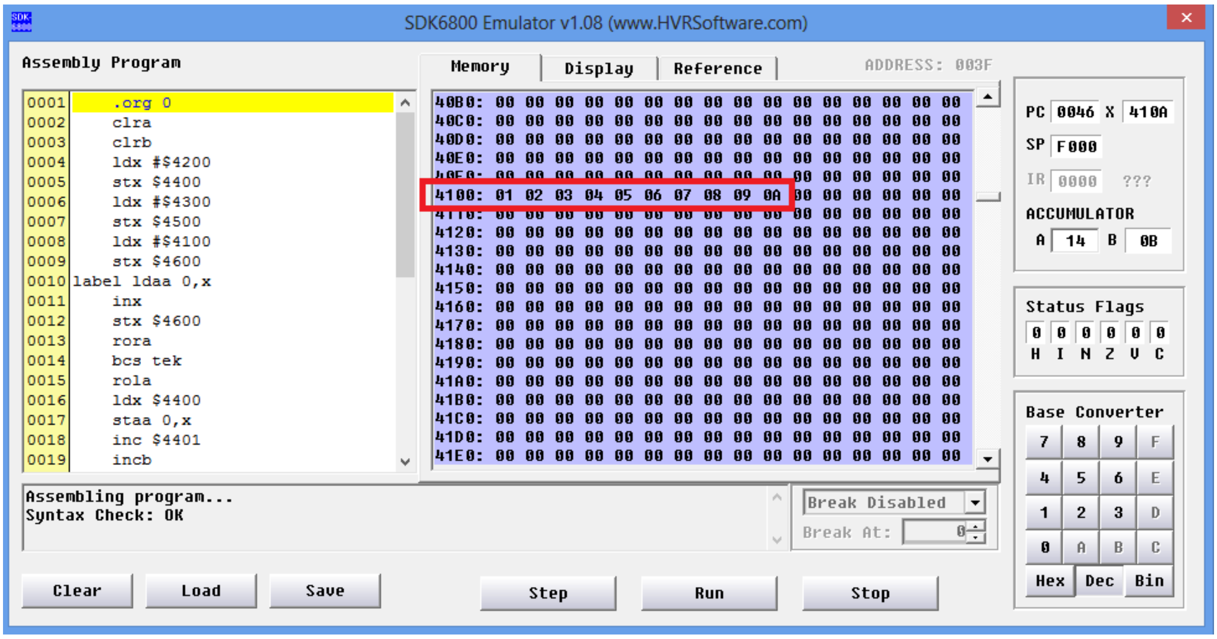
Break At: 0

PC 0046 X 410A
SP F000
IR 0000 ???

ACCUMULATOR
A 14 B 0B

Status Flags

H	I	N	Z	V	C
0	0	0	0	0	0



```
      .org 0
      clra
      clrb
      ldx #$4200
      stx $4400
      ldx #$4300
      stx $4500
      ldx #$4100
      stx $4600
label ldaa 0,x
      inx
      stx $4600
      rora
      bcs tek
      rola
      ldx $4400
      staa 0,x
      inc $4401
      incb
```

ADDRESS: 003F

```
40B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
40C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
40D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
40E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
40F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4100: 01 02 03 04 05 06 07 08 09 0A 00 00 00 00 00 00
4110: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4130: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4140: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4150: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4160: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4170: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4180: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

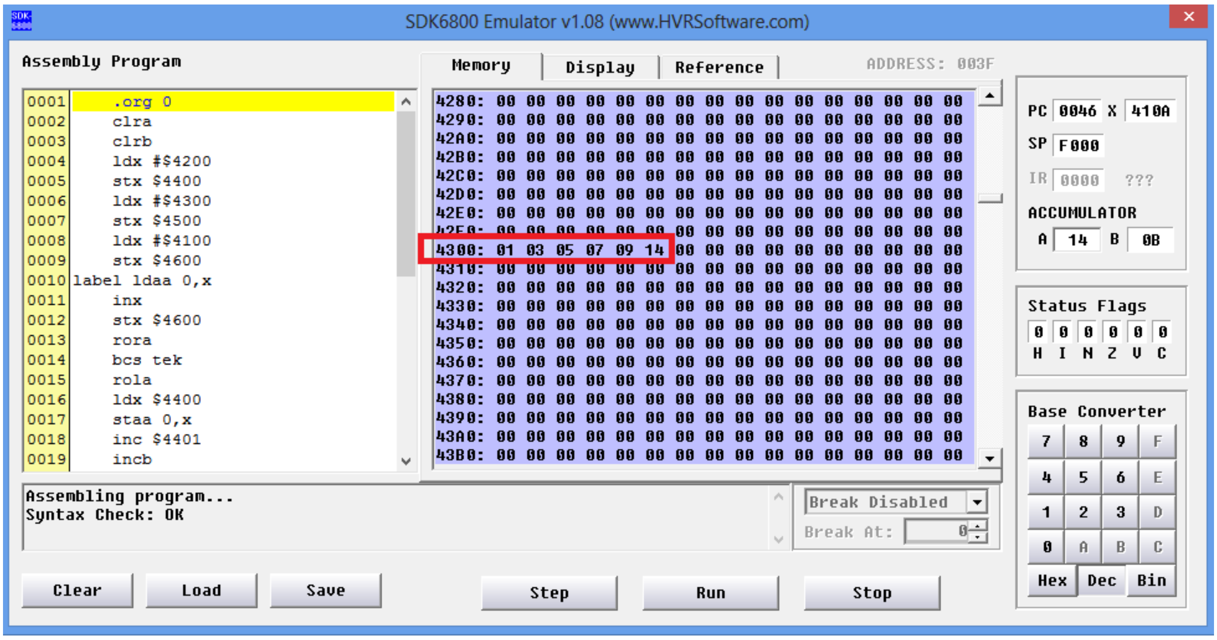
PC 0046 X 410A
SP F000
IR 0000 ???
ACCUMULATOR
A 14 B 0B

Status Flags

0	0	0	0	0	0
H	I	N	Z	V	C

```
Assembling program...
Syntax Check: OK
```

Break Disabled
Break At: 0



```
      .org 0
      clra
      clrb
      ldx #$4200
      stx $4400
      ldx #$4300
      stx $4500
      ldx #$4100
      stx $4600
label ldaa 0,x
      inx
      stx $4600
      rora
      bcs tek
      rola
      ldx $4400
      staa 0,x
      inc $4401
      incb
```

ADDRESS: 003F

```
4280: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4290: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4300: 01 03 05 07 09 14 00 00 00 00 00 00 00 00 00 00
4310: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4320: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4330: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4340: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4350: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4360: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4370: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4380: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4390: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
43A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
43B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

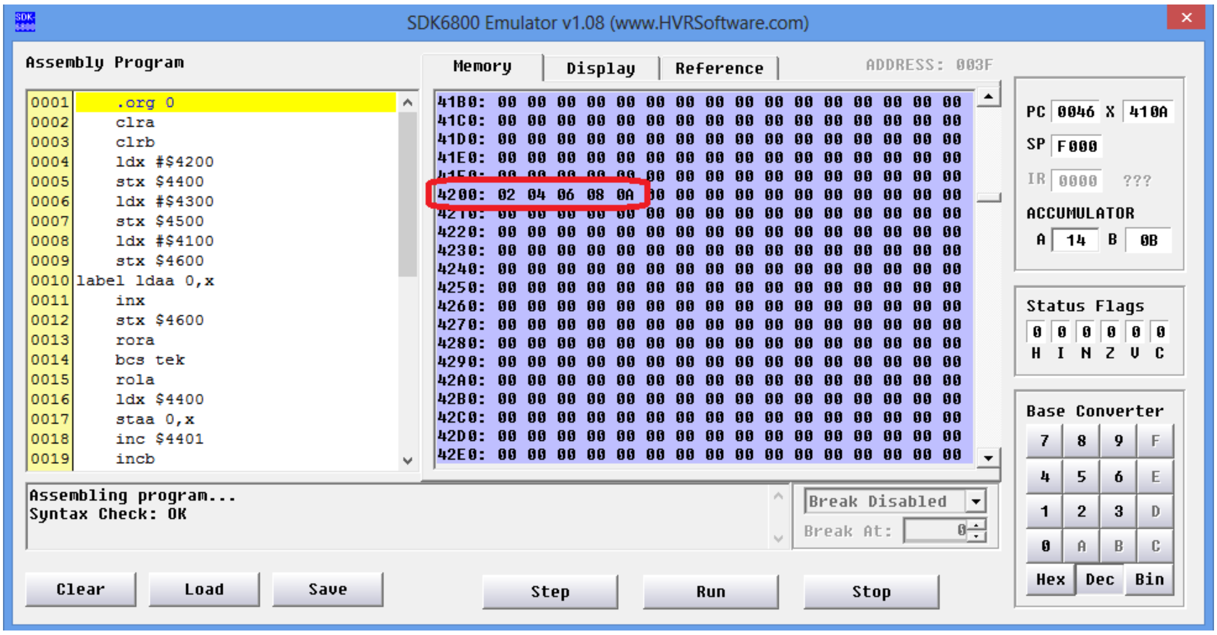
PC 0046 X 410A
SP F000
IR 0000 ???
ACCUMULATOR
A 14 B 0B

Status Flags

0	0	0	0	0	0
H	I	N	Z	V	C

```
Assembling program...
Syntax Check: OK
```

Break Disabled
Break At: 0



Assembly Program

```
      .org 0
      clra
      clrb
      ldx #$4200
      stx $4400
      ldx #$4300
      stx $4500
      ldx #$4100
      stx $4600
label ldaa 0,x
      inx
      stx $4600
      rora
      bcs tek
      rola
      ldx $4400
      staa 0,x
      inc $4401
      incb
```

ADDRESS: 003F

```
41B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
41F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4200: 02 04 06 08 0A 00 00 00 00 00 00 00 00 00 00 00
4210: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4220: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4230: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4240: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4250: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4260: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4270: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4280: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4290: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
42E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

PC 0046 X 410A

SP F000

IR 0000 ???

ACCUMULATOR

A 14 B 0B

Status Flags

H	I	N	Z	V	C
0	0	0	0	0	0

```
Assembling program...
Syntax Check: OK
```

Break Disabled

Break At: 0