# Microprocessor and Computer Architecture (MPCA) Laboratory
# UE20CS252 4th Semester,
# Academic Year 2021-22

| Vanshika Goel | PES1UG20CS484 | H Section |
|---|---|---|

**Week 2 : 04 Feb 2021**

**1) Title of Program:**

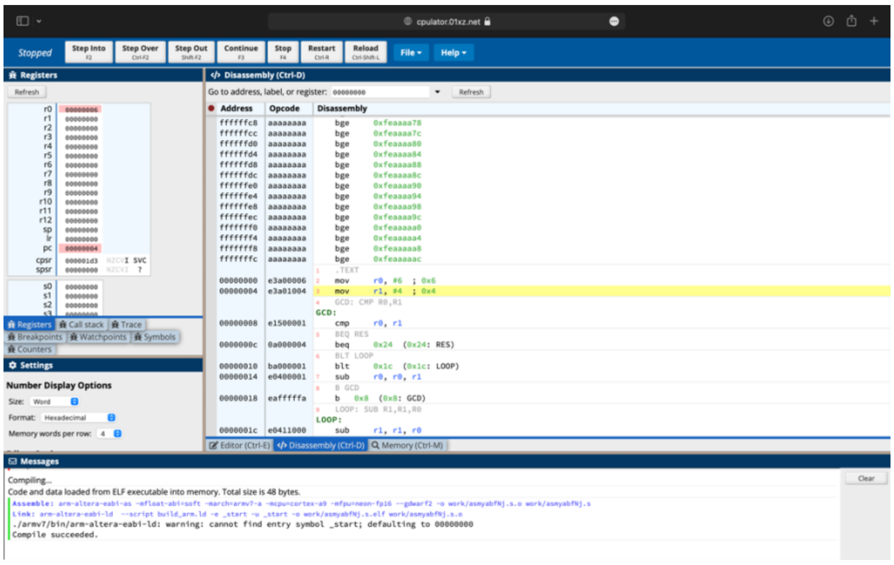
Write a program in ARM7TDMI-ISA to find GCD of two numbers.

a. Assume operands to be in the CPU registers
b. Assume operands in the memory locations.

**Program Code:**
**1)A)**

```
.TEXT
MOV R0,#6
MOV R1,#4
GCD: CMP R0,R1 BEQ RES

BLT LOOP
SUB R0,R0,R1
B GCD
LOOP: SUB R1,R1,R0 B GCD
RES: MOV R2,R0 SWI 0X011
.END
```

**Screenshot of ARMSimulator:**



```
r0    00000006
r1    00000000
r2    00000000
r3    00000000
r4    00000000
r5    00000000
r6    00000000
r7    00000000
r8    00000000
r9    00000000
r10   00000000
r11   00000000
r12   00000000
sp    00000000
lr    00000000
pc    00000004
cpsr  000001d3  NZCVI SVC
spsr  00000000  NZCVI  ?

s0    00000000
s1    00000000
s2    00000000
```

Settings

Number Display Options
Size: Word
Format: Hexadecimal
Memory words per row: 4

Go to address, label, or register: 00000000

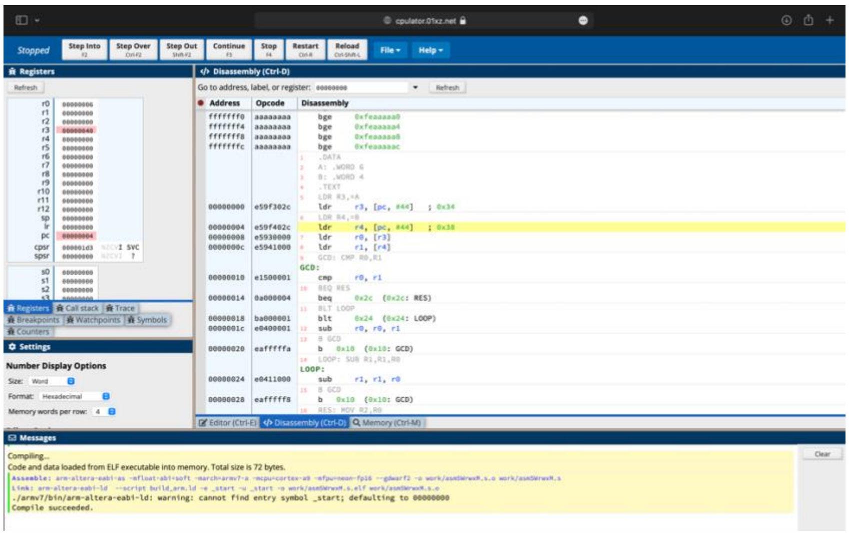
```
Address   Opcode    Disassembly
ffffffc8  aaaaaaaa     bge     0xfeaaaa78
ffffffcc  aaaaaaaa     bge     0xfeaaaa7c
ffffffd0  aaaaaaaa     bge     0xfeaaaa80
ffffffd4  aaaaaaaa     bge     0xfeaaaa84
ffffffd8  aaaaaaaa     bge     0xfeaaaa88
ffffffdc  aaaaaaaa     bge     0xfeaaaa8c
ffffffe0  aaaaaaaa     bge     0xfeaaaa90
ffffffe4  aaaaaaaa     bge     0xfeaaaa94
ffffffe8  aaaaaaaa     bge     0xfeaaaa98
ffffffec  aaaaaaaa     bge     0xfeaaaa9c
fffffff0  aaaaaaaa     bge     0xfeaaaaa0
fffffff4  aaaaaaaa     bge     0xfeaaaaa4
fffffff8  aaaaaaaa     bge     0xfeaaaaa8
fffffffc  aaaaaaaa     bge     0xfeaaaaac
                       .TEXT
00000000  e3a00006     mov     r0, #6   ; 0x6
00000004  e3a01004     mov     r1, #4   ; 0x4
                       GCD: CMP R0,R1
                    GCD:
00000008  e1500001     cmp     r0, r1
                       BEQ RES
0000000c  0a000004     beq     0x24  (0x24: RES)
                       BLT LOOP
00000010  ba000001     blt     0x1c  (0x1c: LOOP)
00000014  e0400001     sub     r0, r0, r1
                       B GCD
00000018  eafffffa     b   0x8  (0x8: GCD)
                       LOOP: SUB R1,R1,R0
                    LOOP:
0000001c  e0411000     sub     r1, r1, r0
```

```
Compiling...
Code and data loaded from ELF executable into memory. Total size is 48 bytes.
Assemble: arm-altera-eabi-as -mfloat-abi=soft -march=armv7-a -mcpu=cortex-a9 -mfpu=neon-fp16 --gdwarf2 -o work/asmyabfNj.s.o work/asmyabfNj.s
Link: arm-altera-eabi-ld  --script build_arm.ld -e _start -u _start -o work/asmyabfNj.s.elf work/asmyabfNj.s.o
./armv7/bin/arm-altera-eabi-ld: warning: cannot find entry symbol _start; defaulting to 00000000
Compile succeeded.
```

**1)B)**

```
.DATA
A: .WORD 6

B: .WORD 4

.TEXT

LDR R3,=A
LDR R4,=B
LDR R0,[R3]
LDR R1,[R4]

GCD: CMP R0,R1

BEQ RES

BLT LOOP
SUB R0,R0,R1
B GCD
LOOP: SUB R1,R1,R0

B GCD
RES: MOV R2,R0

SWI 0X011

.END
```

**Screenshot of ARMSimulator:**



```
r0    00000006
r1    00000000
r2    00000000
r3    00000040
r4    00000000
r5    00000000
r6    00000000
r7    00000000
r8    00000000
r9    00000000
r10   00000000
r11   00000000
r12   00000000
sp    00000000
lr    00000000
pc    00000004
cpsr  000001d3  NZCVI SVC
spsr  00000000  NZCVI  ?

s0    00000000
s1    00000000
s2    00000000
s3    00000000
```

**Number Display Options**

Size: Word

Format: Hexadecimal

Memory words per row: 4

Go to address, label, or register: 00000000

```
Address   Opcode    Disassembly
fffffff0  aaaaaaaa      bge      0xfeaaaaa0
fffffff4  aaaaaaaa      bge      0xfeaaaaa4
fffffff8  aaaaaaaa      bge      0xfeaaaaa8
fffffffc  aaaaaaaa      bge      0xfeaaaaac
                        .DATA
                        A: .WORD 6
                        B: .WORD 4
                        .TEXT
                        LDR R3,=A
00000000  e59f302c      ldr      r3, [pc, #44]    ; 0x34
                        LDR R4,=B
00000004  e59f402c      ldr      r4, [pc, #44]    ; 0x38
00000008  e5930000      ldr      r0, [r3]
0000000c  e5941000      ldr      r1, [r4]
                        GCD: CMP R0,R1
                    GCD:
00000010  e1500001      cmp      r0, r1
                        BEQ RES
00000014  0a000004      beq      0x2c  (0x2c: RES)
                        BLT LOOP
00000018  ba000001      blt      0x24  (0x24: LOOP)
0000001c  e0400001      sub      r0, r0, r1
                        B GCD
00000020  eafffffa      b   0x10  (0x10: GCD)
                        LOOP: SUB R1,R1,R0
                    LOOP:
00000024  e0411000      sub      r1, r1, r0
                        B GCD
00000028  eafffff8      b   0x10  (0x10: GCD)
                        RES: MOV R2,R0
```

```
Compiling...
Code and data loaded from ELF executable into memory. Total size is 72 bytes.
Assemble: arm-altera-eabi-as -mfloat-abi=soft -march=armv7-a -mcpu=cortex-a9 -mfpu=neon-fp16 --gdwarf2 -o work/asm5WrwxM.s.o work/asm5WrwxM.s
Link: arm-altera-eabi-ld  --script build_arm.ld -e _start -u _start -o work/asm5WrwxM.s.elf work/asm5WrwxM.s.o
./armv7/bin/arm-altera-eabi-ld: warning: cannot find entry symbol _start; defaulting to 00000000
Compile succeeded.
```

**2) Title of Program:**

Write a program in ARM7TDMI-ISA to find the sum of N data items in the memory. Store the result in the memory location.

a. Use Pre-indexing addressing mode

b. Use Post- Indexing addressing mode

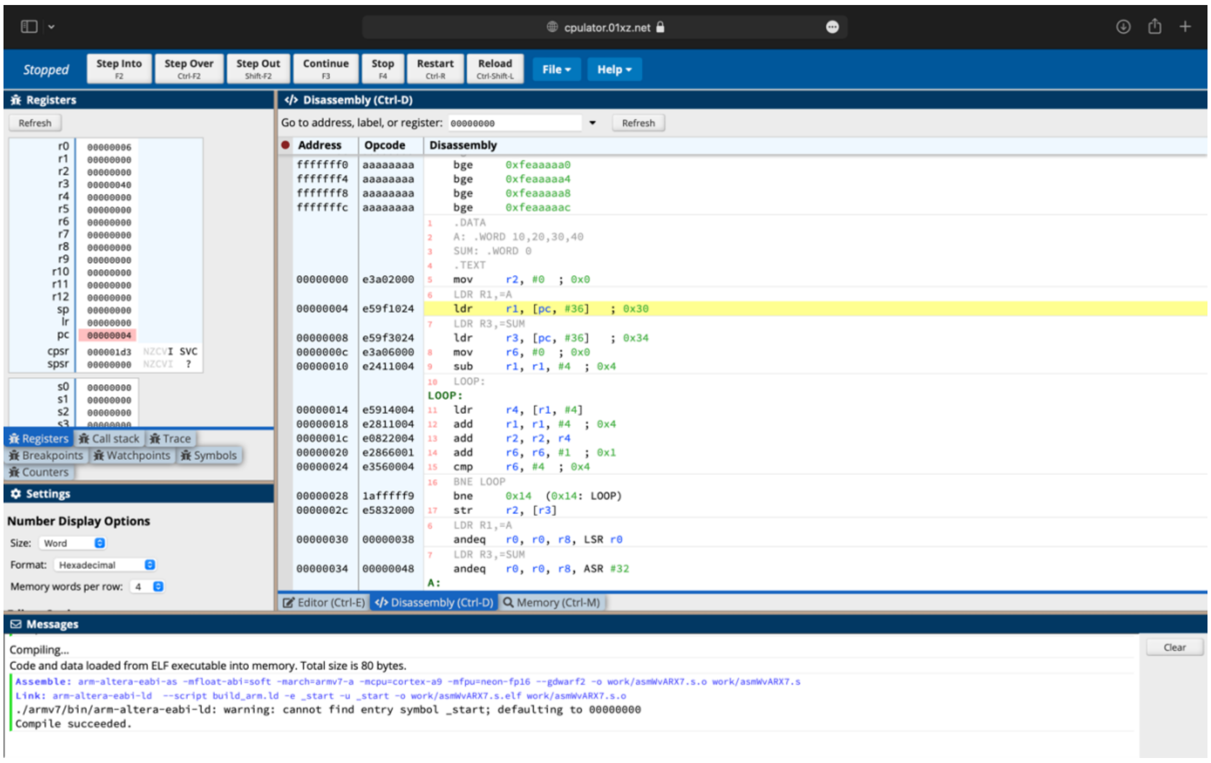
c. Use Auto-indexing addressing mode

**Program Code:**

**2)A)**

```
.DATA
A: .WORD 10,20,30,40 SUM: .WORD 0
.TEXT
MOV R2,#0
LDR R1,=A
LDR R3,=SUM
MOV R6,#0
SUB R1,R1,#4

LOOP:
LDR R4,[R1,#4]

ADD R1,R1,#4

ADD R2,R2,R4

ADD R6,R6,#1

CMP R6,#4

BNE LOOP

STR R2,[R3]

.END
```

**Screenshot of ARMSimulator:**



```
  r0   00000006
  r1   00000000
  r2   00000000
  r3   00000040
  r4   00000000
  r5   00000000
  r6   00000000
  r7   00000000
  r8   00000000
  r9   00000000
 r10   00000000
 r11   00000000
 r12   00000000
  sp   00000000
  lr   00000000
  pc   00000004
cpsr   000001d3  NZCVI SVC
spsr   00000000  NZCVI  ?

  s0   00000000
  s1   00000000
  s2   00000000
  s3   00000000
```

Number Display Options

Size: Word

Format: Hexadecimal

Memory words per row: 4

Go to address, label, or register: 00000000

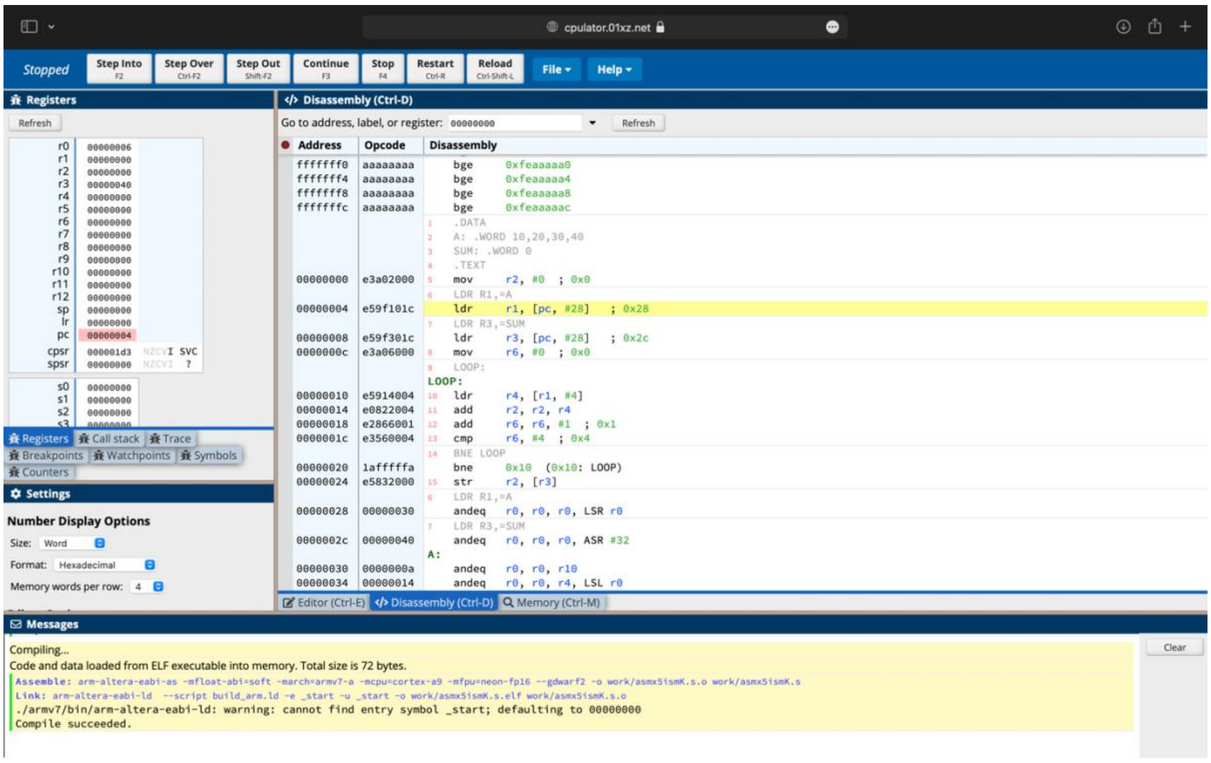
```
Address   Opcode    Disassembly
fffffff0  aaaaaaaa    bge     0xfeaaaaa0
fffffff4  aaaaaaaa    bge     0xfeaaaaa4
fffffff8  aaaaaaaa    bge     0xfeaaaaa8
fffffffc  aaaaaaaa    bge     0xfeaaaaac
                      .DATA
                      A: .WORD 10,20,30,40
                      SUM: .WORD 0
                      .TEXT
00000000  e3a02000    mov     r2, #0  ; 0x0
                      LDR R1,=A
00000004  e59f1024    ldr     r1, [pc, #36]   ; 0x30
                      LDR R3,=SUM
00000008  e59f3024    ldr     r3, [pc, #36]   ; 0x34
0000000c  e3a06000    mov     r6, #0  ; 0x0
00000010  e2411004    sub     r1, r1, #4  ; 0x4
                      LOOP:
                    LOOP:
00000014  e5914004    ldr     r4, [r1, #4]
00000018  e2811004    add     r1, r1, #4  ; 0x4
0000001c  e0822004    add     r2, r2, r4
00000020  e2866001    add     r6, r6, #1  ; 0x1
00000024  e3560004    cmp     r6, #4  ; 0x4
                      BNE LOOP
00000028  1afffff9    bne     0x14  (0x14: LOOP)
0000002c  e5832000    str     r2, [r3]
                      LDR R1,=A
00000030  00000038    andeq   r0, r0, r8, LSR r0
                      LDR R3,=SUM
00000034  00000048    andeq   r0, r0, r8, ASR #32
                    A:
```

```
Compiling...
Code and data loaded from ELF executable into memory. Total size is 80 bytes.
Assemble: arm-altera-eabi-as -mfloat-abi=soft -march=armv7-a -mcpu=cortex-a9 -mfpu=neon-fp16 --gdwarf2 -o work/asmWvARX7.s.o work/asmWvARX7.s
Link: arm-altera-eabi-ld  --script build_arm.ld -e _start -u _start -o work/asmWvARX7.s.elf work/asmWvARX7.s.o
./armv7/bin/arm-altera-eabi-ld: warning: cannot find entry symbol _start; defaulting to 00000000
Compile succeeded.
```

**2)B)**

```
.DATA
A: .WORD 10,20,30,40

SUM: .WORD 0
.TEXT
MOV R2,#0
LDR R1,=A
LDR R3,=SUM
MOV R6,#0

LOOP:
LDR R4,[R1,#4]

ADD R2,R2,R4

ADD R6,R6,#1

CMP R6,#4

BNE LOOP

STR R2,[R3]

.END
```

**Screenshot of ARMSimulator:**



```
r0    00000006
r1    00000000
r2    00000000
r3    00000040
r4    00000000
r5    00000000
r6    00000000
r7    00000000
r8    00000000
r9    00000000
r10   00000000
r11   00000000
r12   00000000
sp    00000000
lr    00000000
pc    00000004
cpsr  000001d3  NZCVI SVC
spsr  00000000  NZCVI  ?
s0    00000000
s1    00000000
s2    00000000
s3    00000000
```

Number Display Options

Size: Word

Format: Hexadecimal

Memory words per row: 4

Go to address, label, or register: 00000000

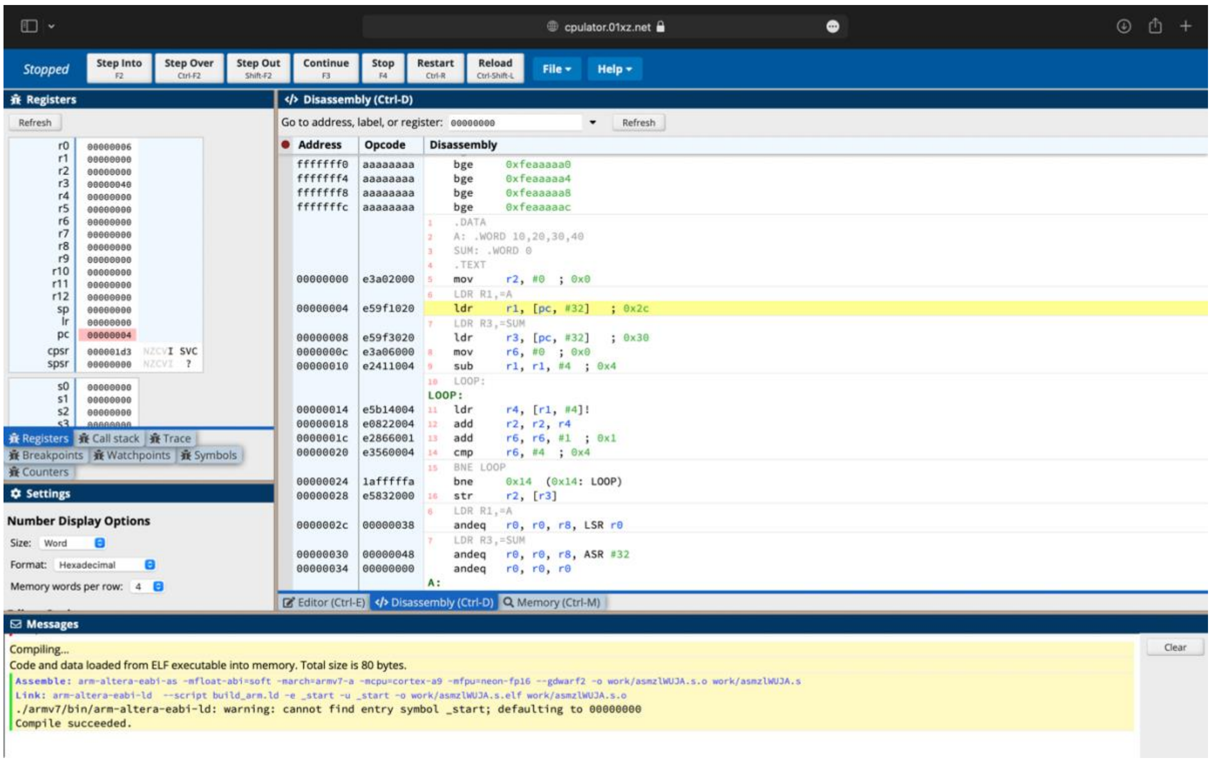
```
Address   Opcode    Disassembly
fffffff0  aaaaaaaa    bge     0xfeaaaaa0
fffffff4  aaaaaaaa    bge     0xfeaaaaa4
fffffff8  aaaaaaaa    bge     0xfeaaaaa8
fffffffc  aaaaaaaa    bge     0xfeaaaaac
                      .DATA
                      A: .WORD 10,20,30,40
                      SUM: .WORD 0
                      .TEXT
00000000  e3a02000    mov     r2, #0  ; 0x0
                      LDR R1,=A
00000004  e59f101c    ldr     r1, [pc, #28]   ; 0x28
                      LDR R3,=SUM
00000008  e59f301c    ldr     r3, [pc, #28]   ; 0x2c
0000000c  e3a06000    mov     r6, #0  ; 0x0
                      LOOP:
                    LOOP:
00000010  e5914004    ldr     r4, [r1, #4]
00000014  e0822004    add     r2, r2, r4
00000018  e2866001    add     r6, r6, #1  ; 0x1
0000001c  e3560004    cmp     r6, #4  ; 0x4
                      BNE LOOP
00000020  1afffffa    bne     0x10  (0x10: LOOP)
00000024  e5832000    str     r2, [r3]
                      LDR R1,=A
00000028  00000030    andeq   r0, r0, r0, LSR r0
                      LDR R3,=SUM
0000002c  00000040    andeq   r0, r0, r0, ASR #32
                    A:
00000030  0000000a    andeq   r0, r0, r10
00000034  00000014    andeq   r0, r0, r4, LSL r0
```

```
Compiling...
Code and data loaded from ELF executable into memory. Total size is 72 bytes.
Assemble: arm-altera-eabi-as -mfloat-abi=soft -march=armv7-a -mcpu=cortex-a9 -mfpu=neon-fp16 --gdwarf2 -o work/asmx5ismK.s.o work/asmx5ismK.s
Link: arm-altera-eabi-ld  --script build_arm.ld -e _start -u _start -o work/asmx5ismK.s.elf work/asmx5ismK.s.o
./armv7/bin/arm-altera-eabi-ld: warning: cannot find entry symbol _start; defaulting to 00000000
Compile succeeded.
```

**2)C)**

```
.DATA
A: .WORD 10,20,30,40 SUM: .WORD 0
.TEXT
MOV R2,#0
LDR R1,=A
LDR R3,=SUM
MOV R6,#0

SUB R1,R1,#4

LOOP:
LDR R4,[R1,#4]!

ADD R2,R2,R4

ADD R6,R6,#1

CMP R6,#4

BNE LOOP

STR R2,[R3]

.END
```

## Screenshot of ARMSimulator:



```
r0    00000006
r1    00000000
r2    00000000
r3    00000040
r4    00000000
r5    00000000
r6    00000000
r7    00000000
r8    00000000
r9    00000000
r10   00000000
r11   00000000
r12   00000000
sp    00000000
lr    00000000
pc    00000004
cpsr  000001d3  NZCVI SVC
spsr  00000000  NZCVI  ?

s0    00000000
s1    00000000
s2    00000000
s3    00000000
```

**Number Display Options**

Size: Word

Format: Hexadecimal

Memory words per row: 4

Go to address, label, or register: 00000000

```
Address   Opcode    Disassembly
fffffff0  aaaaaaaa  bge     0xfeaaaaa0
fffffff4  aaaaaaaa  bge     0xfeaaaaa4
fffffff8  aaaaaaaa  bge     0xfeaaaaa8
fffffffc  aaaaaaaa  bge     0xfeaaaaac
                    .DATA
                    A: .WORD 10,20,30,40
                    SUM: .WORD 0
                    .TEXT
00000000  e3a02000  mov     r2, #0  ; 0x0
                    LDR R1,=A
00000004  e59f1020  ldr     r1, [pc, #32]   ; 0x2c
                    LDR R3,=SUM
00000008  e59f3020  ldr     r3, [pc, #32]   ; 0x30
0000000c  e3a06000  mov     r6, #0  ; 0x0
00000010  e2411004  sub     r1, r1, #4  ; 0x4
                    LOOP:
                  LOOP:
00000014  e5b14004  ldr     r4, [r1, #4]!
00000018  e0822004  add     r2, r2, r4
0000001c  e2866001  add     r6, r6, #1  ; 0x1
00000020  e3560004  cmp     r6, #4  ; 0x4
                    BNE LOOP
00000024  1afffffa  bne     0x14  (0x14: LOOP)
00000028  e5832000  str     r2, [r3]
                    LDR R1,=A
0000002c  00000038  andeq   r0, r0, r8, LSR r0
                    LDR R3,=SUM
00000030  00000048  andeq   r0, r0, r8, ASR #32
00000034  00000000  andeq   r0, r0, r0
                  A:
```

```
Compiling...
Code and data loaded from ELF executable into memory. Total size is 80 bytes.
Assemble: arm-altera-eabi-as -mfloat-abi=soft -march=armv7-a -mcpu=cortex-a9 -mfpu=neon-fp16 --gdwarf2 -o work/asmzlWUJA.s.o work/asmzlWUJA.s
Link: arm-altera-eabi-ld  --script build_arm.ld -e _start -u _start -o work/asmzlWUJA.s.elf work/asmzlWUJA.s.o
./armv7/bin/arm-altera-eabi-ld: warning: cannot find entry symbol _start; defaulting to 00000000
Compile succeeded.
```

**3) Title of Program:**

Write a program in ARM7TDMI-ISA to find the sum of N data items at alternate [ odd or **even** positions] locations in the memory. Store the result in the memory location.

a. Use Pre-indexing addressing mode

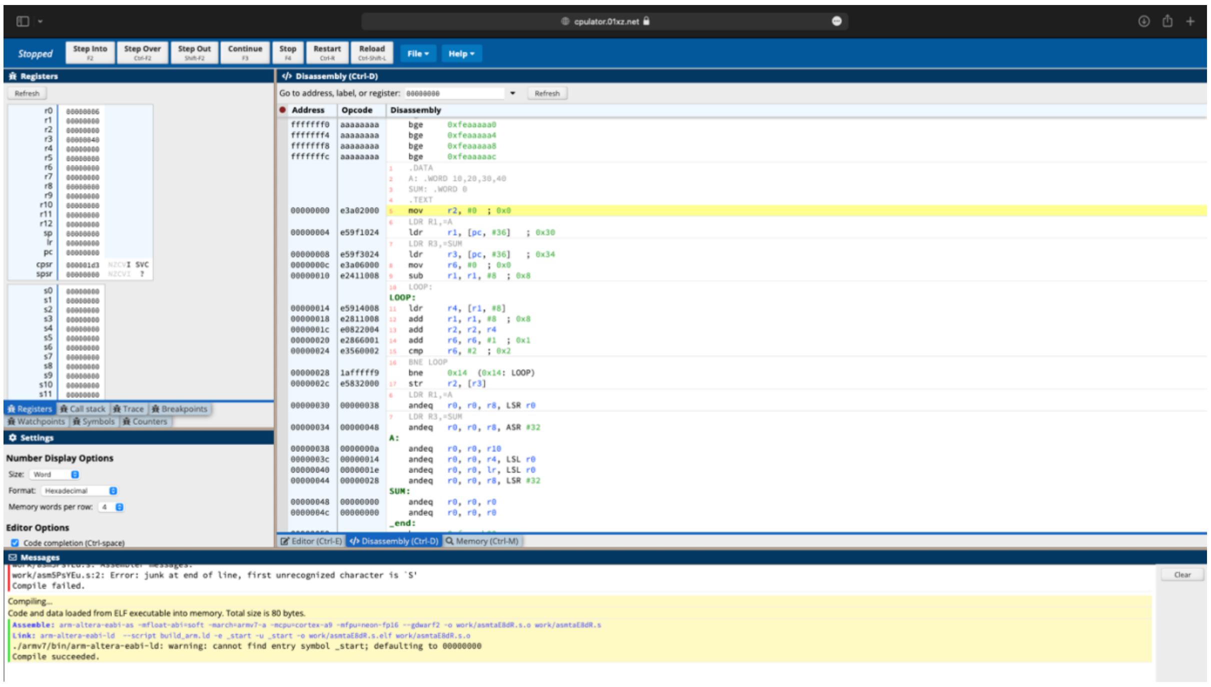
b. Use Post- Indexing addressing mode

c. Use Auto-indexing addressing mode

**Program Code:**
**3)A)**

```
.DATA
A: .WORD 10,20,30,40
SUM: .WORD 0
.TEXT
MOV R2,#0
LDR R1,=A
LDR R3,=SUM
MOV R6,#0
SUB R1,R1,#8
LOOP:
LDR R4,[R1,#8]
ADD R1,R1,#8
ADD R2,R2,R4
ADD R6,R6,#1
CMP R6,#2
BNE LOOP
STR R2,[R3]
.END
```

**Screenshot of ARMSimulator:**



```
r0    00000006
r1    00000000
r2    00000000
r3    00000048
r4    00000000
r5    00000000
r6    00000000
r7    00000000
r8    00000000
r9    00000000
r10   00000000
r11   00000000
r12   00000000
sp    00000000
lr    00000000
pc    00000000
cpsr  000001d3  NZCVI SVC
spsr  00000000  NZCVI  ?

s0    00000000
s1    00000000
s2    00000000
s3    00000000
s4    00000000
s5    00000000
s6    00000000
s7    00000000
s8    00000000
s9    00000000
s10   00000000
s11   00000000
```

Number Display Options
Size: Word
Format: Hexadecimal
Memory words per row: 4

Editor Options
☑ Code completion (Ctrl-space)

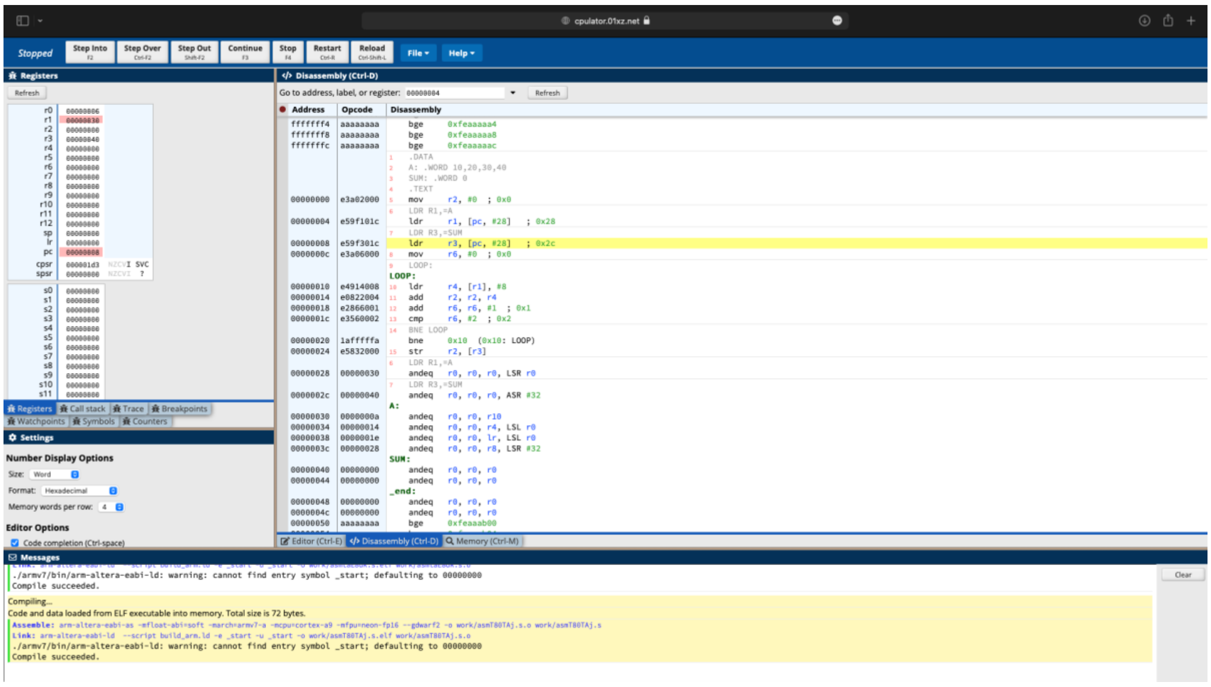
```
Address   Opcode    Disassembly
fffffff0  aaaaaaaa     bge     0xfeaaaaa0
fffffff4  aaaaaaaa     bge     0xfeaaaaa4
fffffff8  aaaaaaaa     bge     0xfeaaaaa8
fffffffc  aaaaaaaa     bge     0xfeaaaaac
                       .DATA
                       A: .WORD 10,20,30,40
                       SUM: .WORD 0
                       .TEXT
00000000  e3a02000     mov     r2, #0  ; 0x0
                       LDR R1,=A
00000004  e59f1024     ldr     r1, [pc, #36]   ; 0x30
                       LDR R3,=SUM
00000008  e59f3024     ldr     r3, [pc, #36]   ; 0x34
0000000c  e3a06000     mov     r6, #0  ; 0x0
00000010  e2411008     sub     r1, r1, #8  ; 0x8
                       LOOP:
                    LOOP:
00000014  e5914008     ldr     r4, [r1, #8]
00000018  e2811008     add     r1, r1, #8  ; 0x8
0000001c  e0822004     add     r2, r2, r4
00000020  e2866001     add     r6, r6, #1  ; 0x1
00000024  e3560002     cmp     r6, #2  ; 0x2
                       BNE LOOP
00000028  1afffff9     bne     0x14  (0x14: LOOP)
0000002c  e5832000     str     r2, [r3]
                       LDR R1,=A
00000030  00000038     andeq   r0, r0, r8, LSR r0
                       LDR R3,=SUM
00000034  00000048     andeq   r0, r0, r8, ASR #32
                    A:
00000038  0000000a     andeq   r0, r0, r10
0000003c  00000014     andeq   r0, r0, r4, LSL r0
00000040  0000001e     andeq   r0, r0, lr, LSL r0
00000044  00000028     andeq   r0, r0, r8, LSR #32
                    SUM:
00000048  00000000     andeq   r0, r0, r0
0000004c  00000000     andeq   r0, r0, r0
                    _end:
```

```
work/asm5PsYEu.s:2: Error: junk at end of line, first unrecognized character is `S'
Compile failed.
Compiling...
Code and data loaded from ELF executable into memory. Total size is 80 bytes.
Assemble: arm-altera-eabi-as -mfloat-abi=soft -march=armv7-a -mcpu=cortex-a9 -mfpu=neon-fp16 --gdwarf2 -o work/asmtaE8dR.s.o work/asmtaE8dR.s
Link: arm-altera-eabi-ld  --script build_arm.ld -e _start -u _start -o work/asmtaE8dR.s.elf work/asmtaE8dR.s.o
./armv7/bin/arm-altera-eabi-ld: warning: cannot find entry symbol _start; defaulting to 00000000
Compile succeeded.
```

**3)B)**

```
.DATA
A: .WORD 10,20,30,40
SUM: .WORD 0
.TEXT
MOV R2,#0
LDR R1,=A
LDR R3,=SUM
MOV R6,#0
LOOP:
LDR R4,[R1],#8
ADD R2,R2,R4
ADD R6,R6,#1
CMP R6,#2
BNE LOOP
STR R2,[R3]
.END
```

**Screenshot of ARMSimulator:**



```
r0    00000000
r1    00000030
r2    00000000
r3    00000040
r4    00000000
r5    00000000
r6    00000000
r7    00000000
r8    00000000
r9    00000000
r10   00000000
r11   00000000
r12   00000000
sp    00000000
lr    00000000
pc    00000008
cpsr  000001d3  NZCVI SVC
spsr  00000000  NZCVI  ?

s0    00000000
s1    00000000
s2    00000000
s3    00000000
s4    00000000
s5    00000000
s6    00000000
s7    00000000
s8    00000000
s9    00000000
s10   00000000
s11   00000000
```

Settings

Number Display Options

Size: Word

Format: Hexadecimal

Memory words per row: 4

Editor Options

☑ Code completion (Ctrl-space)

Go to address, label, or register: 00000004

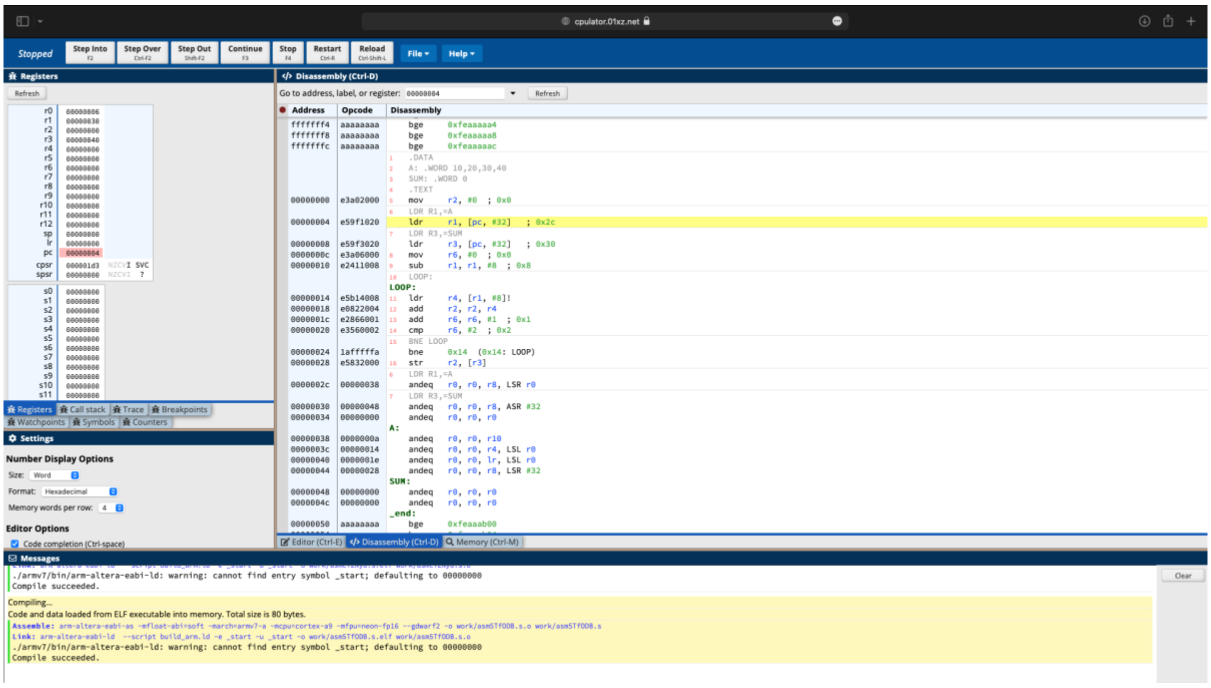
```
Address   Opcode    Disassembly
fffffff4  aaaaaaaa      bge     0xfeaaaaa4
fffffff8  aaaaaaaa      bge     0xfeaaaaa8
fffffffc  aaaaaaaa      bge     0xfeaaaaac
                        .DATA
                        A: .WORD 10,20,30,40
                        SUM: .WORD 0
                        .TEXT
00000000  e3a02000      mov     r2, #0  ; 0x0
                        LDR R1,=A
00000004  e59f101c      ldr     r1, [pc, #28]   ; 0x28
                        LDR R3,=SUM
00000008  e59f301c      ldr     r3, [pc, #28]   ; 0x2c
0000000c  e3a06000      mov     r6, #0  ; 0x0
                        LOOP:
                    LOOP:
00000010  e4914008      ldr     r4, [r1], #8
00000014  e0822004      add     r2, r2, r4
00000018  e2866001      add     r6, r6, #1  ; 0x1
0000001c  e3560002      cmp     r6, #2  ; 0x2
                        BNE LOOP
00000020  1afffffa      bne     0x10  (0x10: LOOP)
00000024  e5832000      str     r2, [r3]
                        LDR R1,=A
00000028  00000030      andeq   r0, r0, r0, LSR r0
                        LDR R3,=SUM
0000002c  00000040      andeq   r0, r0, r0, ASR #32
                    A:
00000030  0000000a      andeq   r0, r0, r10
00000034  00000014      andeq   r0, r0, r4, LSL r0
00000038  0000001e      andeq   r0, r0, lr, LSL r0
0000003c  00000028      andeq   r0, r0, r8, LSR #32
                    SUM:
00000040  00000000      andeq   r0, r0, r0
00000044  00000000      andeq   r0, r0, r0
                    _end:
00000048  00000000      andeq   r0, r0, r0
0000004c  00000000      andeq   r0, r0, r0
00000050  aaaaaaaa      bge     0xfeaaab00
```

```
./armv7/bin/arm-altera-eabi-ld: warning: cannot find entry symbol _start; defaulting to 00000000
Compile succeeded.
Compiling...
Code and data loaded from ELF executable into memory. Total size is 72 bytes.
Assemble: arm-altera-eabi-as -mfloat-abi=soft -march=armv7-a -mcpu=cortex-a9 -mfpu=neon-fp16 --gdwarf2 -o work/asmT80TAj.s.o work/asmT80TAj.s
Link: arm-altera-eabi-ld  --script build_arm.ld -e _start -u _start -o work/asmT80TAj.s.elf work/asmT80TAj.s.o
./armv7/bin/arm-altera-eabi-ld: warning: cannot find entry symbol _start; defaulting to 00000000
Compile succeeded.
```

**3)C)**

```
.DATA
A: .WORD 10,20,30,40
SUM: .WORD 0
.TEXT
MOV R2,#0
LDR R1,=A
LDR R3,=SUM
MOV R6,#0
SUB R1,R1,#8
LOOP:
LDR R4,[R1,#8]!
ADD R2,R2,R4
ADD R6,R6,#1
CMP R6,#2
BNE LOOP
STR R2,[R3]
.END
```

**Screenshot of ARMSimulator:**



```
r0    00000006
r1    00000030
r2    00000000
r3    00000040
r4    00000000
r5    00000000
r6    00000000
r7    00000000
r8    00000000
r9    00000000
r10   00000000
r11   00000000
r12   00000000
sp    00000000
lr    00000000
pc    00000004
cpsr  000001d3  NZCVI SVC
spsr  00000000  NZCVI  ?

s0    00000000
s1    00000000
s2    00000000
s3    00000000
s4    00000000
s5    00000000
s6    00000000
s7    00000000
s8    00000000
s9    00000000
s10   00000000
s11   00000000
```

Settings

Number Display Options

Size: Word

Format: Hexadecimal

Memory words per row: 4

Editor Options

☑ Code completion (Ctrl-space)

Go to address, label, or register: 00000004

```
Address   Opcode    Disassembly
fffffff4  aaaaaaaa     bge     0xfeaaaaa4
fffffff8  aaaaaaaa     bge     0xfeaaaaa8
fffffffc  aaaaaaaa     bge     0xfeaaaaac
                       .DATA
                       A: .WORD 10,20,30,40
                       SUM: .WORD 0
                       .TEXT
00000000  e3a02000     mov     r2, #0  ; 0x0
                       LDR R1,=A
00000004  e59f1020     ldr     r1, [pc, #32]   ; 0x2c
                       LDR R3,=SUM
00000008  e59f3020     ldr     r3, [pc, #32]   ; 0x30
0000000c  e3a06000     mov     r6, #0  ; 0x0
00000010  e2411008     sub     r1, r1, #8  ; 0x8
                       LOOP:
                    LOOP:
00000014  e5b14008     ldr     r4, [r1, #8]!
00000018  e0822004     add     r2, r2, r4
0000001c  e2866001     add     r6, r6, #1  ; 0x1
00000020  e3560002     cmp     r6, #2  ; 0x2
                       BNE LOOP
00000024  1afffffa     bne     0x14  (0x14: LOOP)
00000028  e5832000     str     r2, [r3]
                       LDR R1,=A
0000002c  00000038     andeq   r0, r0, r8, LSR r0
                       LDR R3,=SUM
00000030  00000048     andeq   r0, r0, r8, ASR #32
00000034  00000000     andeq   r0, r0, r0
                    A:
00000038  0000000a     andeq   r0, r0, r10
0000003c  00000014     andeq   r0, r0, r4, LSL r0
00000040  0000001e     andeq   r0, r0, lr, LSL r0
00000044  00000028     andeq   r0, r0, r8, LSR #32
                    SUM:
00000048  00000000     andeq   r0, r0, r0
0000004c  00000000     andeq   r0, r0, r0
                    _end:
00000050  aaaaaaaa     bge     0xfeaaab00
```

```
./armv7/bin/arm-altera-eabi-ld: warning: cannot find entry symbol _start; defaulting to 00000000
Compile succeeded.

Compiling...
Code and data loaded from ELF executable into memory. Total size is 80 bytes.
Assemble: arm-altera-eabi-as -mfloat-abi=soft -march=armv7-a -mcpu=cortex-a9 -mfpu=neon-fp16 --gdwarf2 -o work/asm5TfOD8.s.o work/asm5TfOD8.s
Link: arm-altera-eabi-ld  --script build_arm.ld -e _start -u _start -o work/asm5TfOD8.s.elf work/asm5TfOD8.s.o
./armv7/bin/arm-altera-eabi-ld: warning: cannot find entry symbol _start; defaulting to 00000000
Compile succeeded.
```

**4) Title of Program:**

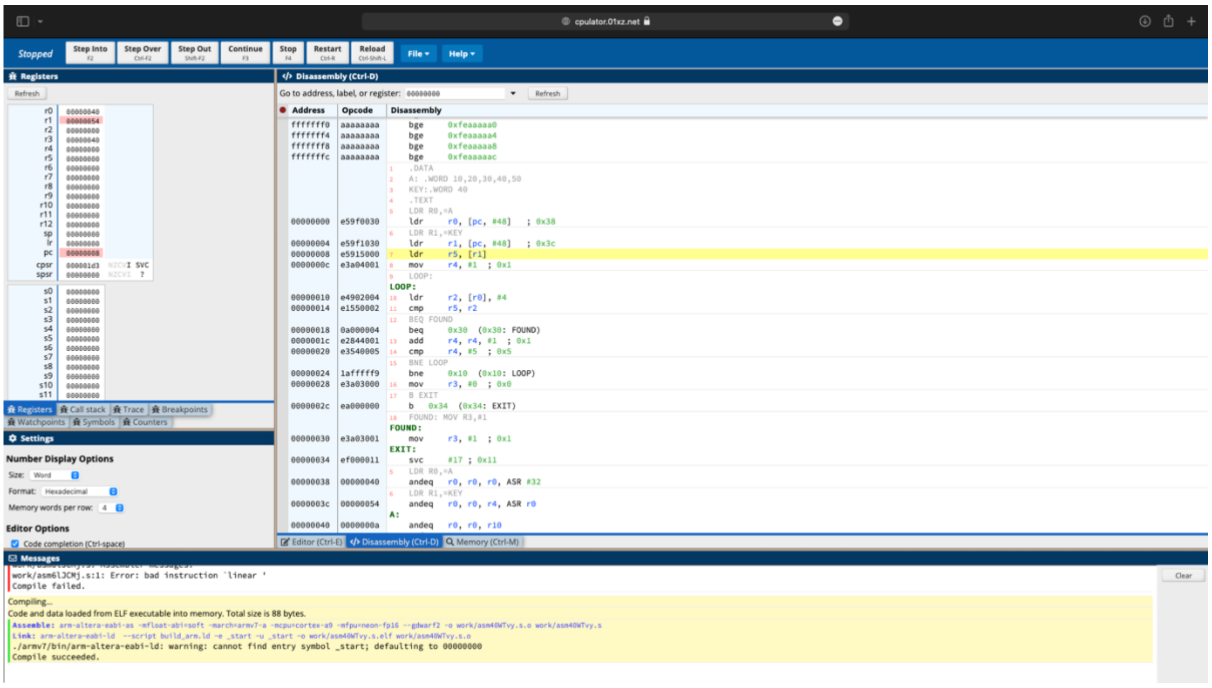
Write a program in ARM7TDMI-ISA to search for an element in an array. Store 00 if the search is unsuccessful and 01 if the search is successful in the register.

a. Use Linear Search Technique

**Program Code:**

```
.DATA
A: .WORD 10,20,30,40,50
KEY:.WORD 40
.TEXT
LDR R0,=A
LDR R1,=KEY
LDR R5,[R1]
MOV R4,#1
LOOP:
LDR R2,[R0],#4
CMP R5,R2
BEQ FOUND
ADD R4,R4,#1
CMP R4,#5
BNE LOOP
MOV R3,#0
B EXIT
FOUND: MOV R3,#1
EXIT:SWI 0X011
```

**Screenshot of ARMSimulator:**



```
r0    00000040
r1    00000054
r2    00000000
r3    00000040
r4    00000000
r5    00000000
r6    00000000
r7    00000000
r8    00000000
r9    00000000
r10   00000000
r11   00000000
r12   00000000
sp    00000000
lr    00000000
pc    00000008
cpsr  000001d3  NZCVI SVC
spsr  00000000  NZCVI ?

s0    00000000
s1    00000000
s2    00000000
s3    00000000
s4    00000000
s5    00000000
s6    00000000
s7    00000000
s8    00000000
s9    00000000
s10   00000000
s11   00000000
```

Number Display Options

Size: Word

Format: Hexadecimal

Memory words per row: 4

Editor Options

☑ Code completion (Ctrl-space)

Go to address, label, or register: 00000000

```
Address   Opcode    Disassembly
fffffff0  aaaaaaaa      bge     0xfeaaaaa0
fffffff4  aaaaaaaa      bge     0xfeaaaaa4
fffffff8  aaaaaaaa      bge     0xfeaaaaa8
fffffffc  aaaaaaaa      bge     0xfeaaaaac
                        .DATA
                        A: .WORD 10,20,30,40,50
                        KEY:.WORD 40
                        .TEXT
                        LDR R0,=A
00000000  e59f0030      ldr     r0, [pc, #48]    ; 0x38
                        LDR R1,=KEY
00000004  e59f1030      ldr     r1, [pc, #48]    ; 0x3c
00000008  e5915000      ldr     r5, [r1]
0000000c  e3a04001      mov     r4, #1  ; 0x1
                        LOOP:
                    LOOP:
00000010  e4902004      ldr     r2, [r0], #4
00000014  e1550002      cmp     r5, r2
                        BEQ FOUND
00000018  0a000004      beq     0x30  (0x30: FOUND)
0000001c  e2844001      add     r4, r4, #1  ; 0x1
00000020  e3540005      cmp     r4, #5  ; 0x5
                        BNE LOOP
00000024  1afffff9      bne     0x10  (0x10: LOOP)
00000028  e3a03000      mov     r3, #0  ; 0x0
                        B EXIT
0000002c  ea000000      b   0x34  (0x34: EXIT)
                        FOUND: MOV R3,#1
                    FOUND:
00000030  e3a03001      mov     r3, #1  ; 0x1
                    EXIT:
00000034  ef000011      svc     #17 ; 0x11
                        LDR R0,=A
00000038  00000040      andeq   r0, r0, r0, ASR #32
                        LDR R1,=KEY
0000003c  00000054      andeq   r0, r0, r4, ASR r0
                    A:
00000040  0000000a      andeq   r0, r0, r10
```

```
work/asm6lJCMj.s:1: Error: bad instruction `linear '
Compile failed.
```

```
Compiling...
Code and data loaded from ELF executable into memory. Total size is 88 bytes.
Assemble: arm-altera-eabi-as -mfloat-abi=soft -march=armv7-a -mcpu=cortex-a9 -mfpu=neon-fp16 --gdwarf2 -o work/asm40WTvy.s.o work/asm40WTvy.s
Link: arm-altera-eabi-ld  --script build_arm.ld -e _start -u _start -o work/asm40WTvy.s.elf work/asm40WTvy.s.o
./armv7/bin/arm-altera-eabi-ld: warning: cannot find entry symbol _start; defaulting to 00000000
Compile succeeded.
```