Faculty of Science, Engineering and Technology

Computer Systems

Name: Le Quang Hai

Student ID: 104175779

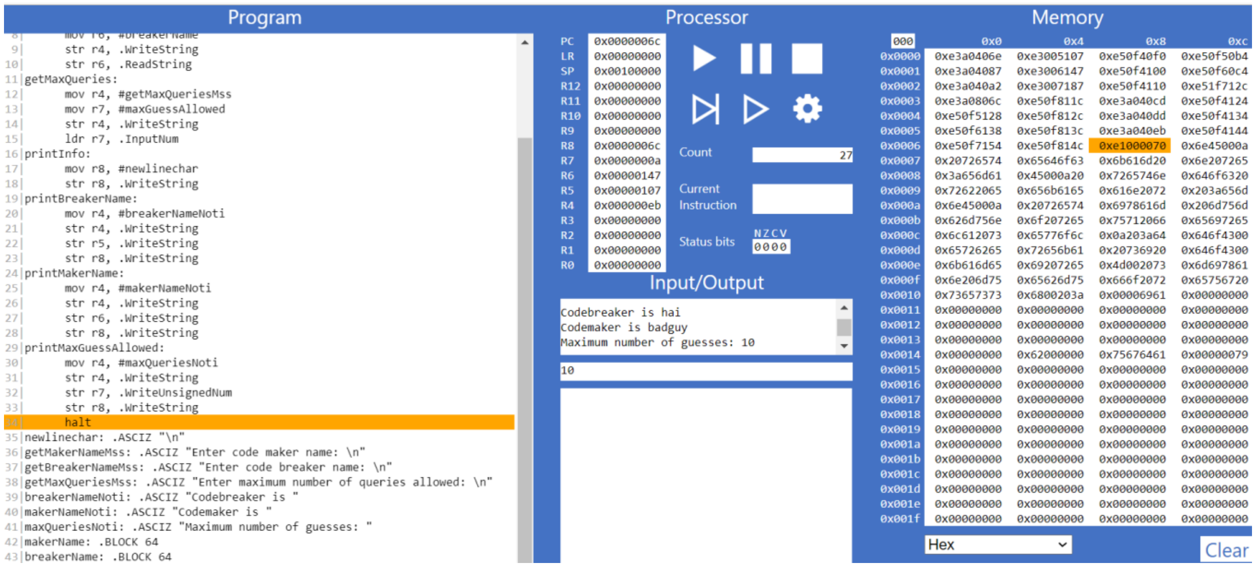
Unit Code: COS10004

Lab Session: Assignment 2

<u>Note:</u> In this piece of work, as instructed, it is assumed that the code entered ONLY contains DIFFERENT color values. Therefore, any test case containing duplicated value (rrrr, ggbb, ppcc,…) should be beyond the scope of the assignment

Step 1: Game set up.

- We use R4 to store the address of the request message and print that on the console, asking for input (breaker name & maker name) and store it then print it



```
        mov r6, #breakerName
        str r4, .WriteString
        str r6, .ReadString
getMaxQueries:
        mov r4, #getMaxQueriesMss
        mov r7, #maxGuessAllowed
        str r4, .WriteString
        ldr r7, .InputNum
printInfo:
        mov r8, #newlinechar
        str r8, .WriteString
printBreakerName:
        mov r4, #breakerNameNoti
        str r4, .WriteString
        str r5, .WriteString
        str r8, .WriteString
printMakerName:
        mov r4, #makerNameNoti
        str r4, .WriteString
        str r6, .WriteString
        str r8, .WriteString
printMaxGuessAllowed:
        mov r4, #maxQueriesNoti
        str r4, .WriteString
        str r7, .WriteUnsignedNum
        str r8, .WriteString
        halt
newlinechar: .ASCIZ "\n"
getMakerNameMss: .ASCIZ "Enter code maker name: \n"
getBreakerNameMss: .ASCIZ "Enter code breaker name: \n"
getMaxQueriesMss: .ASCIZ "Enter maximum number of queries allowed: \n"
breakerNameNoti: .ASCIZ "Codebreaker is "
makerNameNoti: .ASCIZ "Codemaker is "
maxQueriesNoti: .ASCIZ "Maximum number of guesses: "
makerName: .BLOCK 64
breakerName: .BLOCK 64
```

```
PC   0x0000006c
LR   0x00000000
SP   0x00100000
R12  0x00000000
R11  0x00000000
R10  0x00000000
R9   0x00000000
R8   0x0000006c
R7   0x0000000a
R6   0x00000147
R5   0x00000107
R4   0x000000eb
R3   0x00000000
R2   0x00000000
R1   0x00000000
R0   0x00000000

Count 27
Status bits NZCV 0000
```

```
Codebreaker is hai
Codemaker is badguy
Maximum number of guesses: 10

10
```

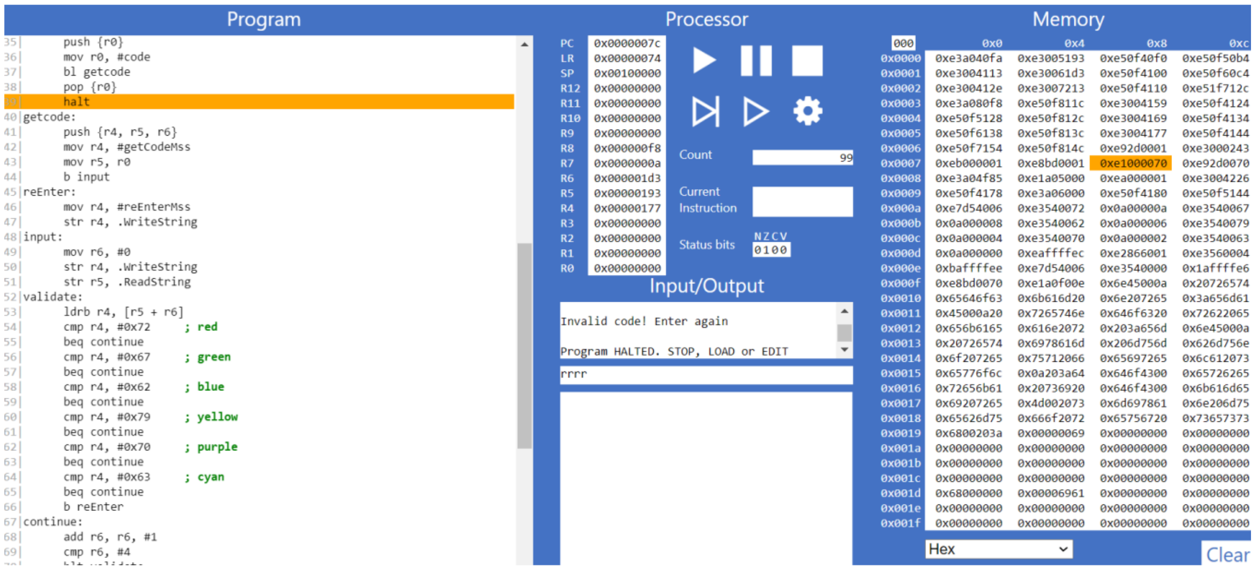
```
000            0x0         0x4         0x8         0xc
0x0000  0xe3a0406e  0xe3005107  0xe50f40f0  0xe50f50b4
0x0001  0xe3a04087  0xe3006147  0xe50f4100  0xe50f60c4
0x0002  0xe3a040a2  0xe3007187  0xe50f4110  0xe51f712c
0x0003  0xe3a0806c  0xe50f811c  0xe3a040cd  0xe50f4124
0x0004  0xe50f5128  0xe50f812c  0xe3a040dd  0xe50f4134
0x0005  0xe50f6138  0xe50f813c  0xe3a040eb  0xe50f4144
0x0006  0xe50f7154  0xe50f814c  0xe1000070  0x6e45000a
0x0007  0x20726574  0x65646f63  0x6b616d20  0x6e207265
0x0008  0x3a656d61  0x45000a20  0x7265746e  0x646f6320
0x0009  0x72622065  0x656b6165  0x616e2072  0x203a656d
0x000a  0x6e45000a  0x20726574  0x6978616d  0x206d756d
0x000b  0x626d756e  0x6f207265  0x75712066  0x65697265
0x000c  0x6c612073  0x65776f6c  0x0a203a64  0x646f4300
0x000d  0x65726265  0x72656b61  0x20736920  0x646f4300
0x000e  0x6b616d65  0x69207265  0x4d002073  0x6d697861
0x000f  0x6e206d75  0x65626d75  0x666f2072  0x65756720
0x0010  0x73657373  0x6800203a  0x00006961  0x00000000
0x0011  0x00000000  0x00000000  0x00000000  0x00000000
0x0012  0x00000000  0x00000000  0x00000000  0x00000000
0x0013  0x00000000  0x00000000  0x00000000  0x00000000
0x0014  0x00000000  0x62000000  0x75676461  0x00000079
0x0015  0x00000000  0x00000000  0x00000000  0x00000000
0x0016  0x00000000  0x00000000  0x00000000  0x00000000
0x0017  0x00000000  0x00000000  0x00000000  0x00000000
0x0018  0x00000000  0x00000000  0x00000000  0x00000000
0x0019  0x00000000  0x00000000  0x00000000  0x00000000
0x001a  0x00000000  0x00000000  0x00000000  0x00000000
0x001b  0x00000000  0x00000000  0x00000000  0x00000000
0x001c  0x00000000  0x00000000  0x00000000  0x00000000
0x001d  0x00000000  0x00000000  0x00000000  0x00000000
0x001e  0x00000000  0x00000000  0x00000000  0x00000000
0x001f  0x00000000  0x00000000  0x00000000  0x00000000
```

Step 2: A code entry function

- The ‘getcode’ function takes a parameter which is a string store in address [R0] and then prints the message on the console, waiting for input string which is the code (secret or guess)

- To validate the code entered, we loop through 4 first characters and check if they are ‘r’, ‘g’, ‘b’, ‘y’, ‘p’, ‘c’, or none.

- To check if there are more than 4 characters entered, the ‘code’ is initiated as a 5-element array. We check the fifth element to see if it is 0 (initiated value). If not, there are more than 5 characters entered.



Program

```
        push {r0}
        mov r0, #code
        bl getcode
        pop {r0}
        halt
getcode:
        push {r4, r5, r6}
        mov r4, #getCodeMss
        mov r5, r0
        b input
reEnter:
        mov r4, #reEnterMss
        str r4, .WriteString
input:
        mov r6, #0
        str r4, .WriteString
        str r5, .ReadString
validate:
        ldrb r4, [r5 + r6]
        cmp r4, #0x72     ; red
        beq continue
        cmp r4, #0x67     ; green
        beq continue
        cmp r4, #0x62     ; blue
        beq continue
        cmp r4, #0x79     ; yellow
        beq continue
        cmp r4, #0x70     ; purple
        beq continue
        cmp r4, #0x63     ; cyan
        beq continue
        b reEnter
continue:
        add r6, r6, #1
        cmp r6, #4
```

Processor

```
PC   0x0000007c
LR   0x00000074
SP   0x00100000
R12  0x00000000
R11  0x00000000
R10  0x00000000
R9   0x00000000
R8   0x000000f8
R7   0x0000000a
R6   0x000001d3
R5   0x00000193
R4   0x00000177
R3   0x00000000
R2   0x00000000
R1   0x00000000
R0   0x00000000

Count                99
Current Instruction
Status bits   NZCV
              0100
```

Input/Output

```
Invalid code! Enter again

Program HALTED. STOP, LOAD or EDIT
```

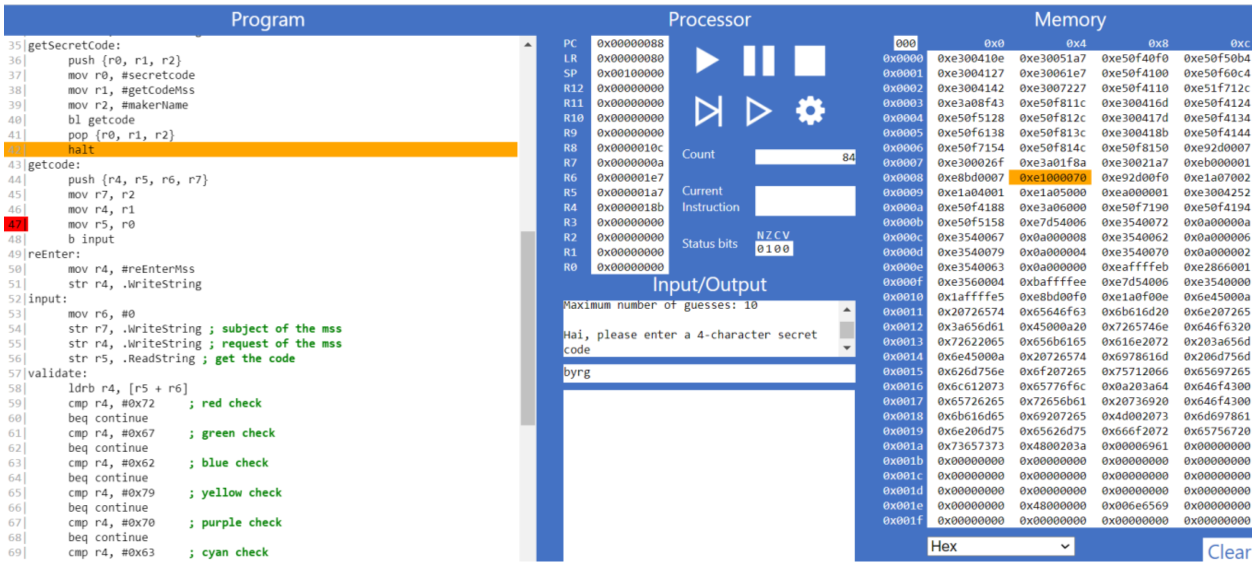
```
rrrr
```

Memory

```
000            0x0         0x4         0x8         0xc
0x0000  0xe3a040fa  0xe3005193  0xe50f40f0  0xe50f50b4
0x0001  0xe3004113  0xe30061d3  0xe50f4100  0xe50f60c4
0x0002  0xe300412e  0xe3007213  0xe50f4110  0xe51f712c
0x0003  0xe3a080f8  0xe50f811c  0xe3004159  0xe50f4124
0x0004  0xe50f5128  0xe50f812c  0xe3004169  0xe50f4134
0x0005  0xe50f6138  0xe50f813c  0xe3004177  0xe50f4144
0x0006  0xe50f7154  0xe50f814c  0xe92d0001  0xe3000243
0x0007  0xeb000001  0xe8bd0001  0xe1000070  0xe92d0070
0x0008  0xe3a04f85  0xe1a05000  0xea000001  0xe3004226
0x0009  0xe50f4178  0xe3a06000  0xe50f4180  0xe50f5144
0x000a  0xe7d54006  0xe3540072  0x0a00000a  0xe3540067
0x000b  0x0a000008  0xe3540062  0x0a000006  0xe3540079
0x000c  0x0a000004  0xe3540070  0x0a000002  0xe3540063
0x000d  0x0a000000  0xeaffffec  0xe2866001  0xe3560004
0x000e  0xbaffffee  0xe7d54006  0xe3540000  0x1affffe6
0x000f  0xe8bd0070  0xe1a0f00e  0x6e45000a  0x20726574
0x0010  0x65646f63  0x6b616d20  0x6e207265  0x3a656d61
0x0011  0x45000a20  0x7265746e  0x646f6320  0x72622065
0x0012  0x656b6165  0x616e2072  0x203a656d  0x6e45000a
0x0013  0x20726574  0x6978616d  0x206d756d  0x626d756e
0x0014  0x6f207265  0x75712066  0x65697265  0x6c612073
0x0015  0x65776f6c  0x0a203a64  0x646f4300  0x65726265
0x0016  0x72656b61  0x20736920  0x646f4300  0x6b616d65
0x0017  0x69207265  0x4d002073  0x6d697861  0x6e206d75
0x0018  0x65626d75  0x666f2072  0x65756720  0x73657373
0x0019  0x6800203a  0x00000069  0x00000000  0x00000000
0x001a  0x00000000  0x00000000  0x00000000  0x00000000
0x001b  0x00000000  0x00000000  0x00000000  0x00000000
0x001c  0x00000000  0x00000000  0x00000000  0x00000000
0x001d  0x68000000  0x00006961  0x00000000  0x00000000
0x001e  0x00000000  0x00000000  0x00000000  0x00000000
0x001f  0x00000000  0x00000000  0x00000000  0x00000000
```

Step 3: Getting the secret code

- We make use of the ‘getcode’ function above with message “Code maker name, please enter a 4-character secret code”



Program

```
getSecretCode:
        push {r0, r1, r2}
        mov r0, #secretcode
        mov r1, #getCodeMss
        mov r2, #makerName
        bl getcode
        pop {r0, r1, r2}
        halt
getcode:
        push {r4, r5, r6, r7}
        mov r7, r2
        mov r4, r1
        mov r5, r0
        b input
reEnter:
        mov r4, #reEnterMss
        str r4, .WriteString
input:
        mov r6, #0
        str r7, .WriteString ; subject of the mss
        str r4, .WriteString ; request of the mss
        str r5, .ReadString ; get the code
validate:
        ldrb r4, [r5 + r6]
        cmp r4, #0x72     ; red check
        beq continue
        cmp r4, #0x67     ; green check
        beq continue
        cmp r4, #0x62     ; blue check
        beq continue
        cmp r4, #0x79     ; yellow check
        beq continue
        cmp r4, #0x70     ; purple check
        beq continue
        cmp r4, #0x63     ; cyan check
```

Processor

```
PC   0x00000088
LR   0x00000080
SP   0x00100000
R12  0x00000000
R11  0x00000000
R10  0x00000000
R9   0x00000000
R8   0x0000010c
R7   0x0000000a
R6   0x000001e7
R5   0x000001a7
R4   0x0000018b
R3   0x00000000
R2   0x00000000
R1   0x00000000
R0   0x00000000

Count                84
Current Instruction
Status bits   NZCV
              0100
```

Input/Output

```
Maximum number of guesses: 10

Hai, please enter a 4-character secret
code
```

```
byrg
```

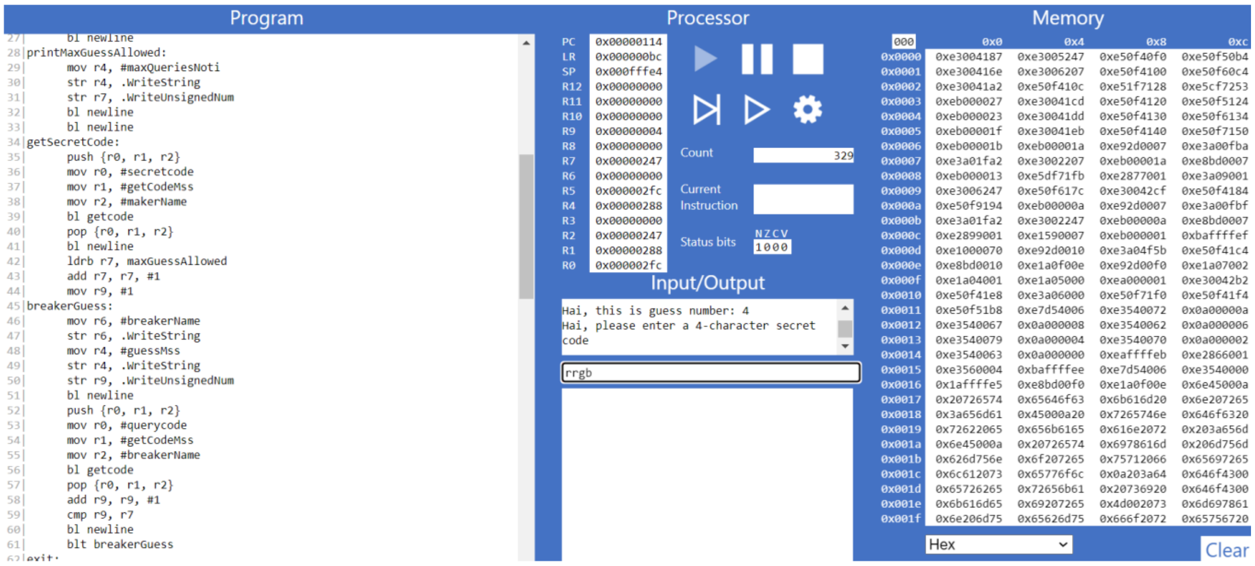
Memory

```
000            0x0         0x4         0x8         0xc
0x0000  0xe300410e  0xe30051a7  0xe50f40f0  0xe50f50b4
0x0001  0xe3004127  0xe30061e7  0xe50f4100  0xe50f60c4
0x0002  0xe3004142  0xe3007227  0xe50f4110  0xe51f712c
0x0003  0xe3a08f43  0xe50f811c  0xe300416d  0xe50f4124
0x0004  0xe50f5128  0xe50f812c  0xe300417d  0xe50f4134
0x0005  0xe50f6138  0xe50f813c  0xe300418b  0xe50f4144
0x0006  0xe50f7154  0xe50f814c  0xe50f8150  0xe92d0007
0x0007  0xe300026f  0xe3a01f8a  0xe30021a7  0xeb000001
0x0008  0xe8bd0007  0xe1000070  0xe92d00f0  0xe1a07002
0x0009  0xe1a04001  0xe1a05000  0xea000001  0xe3004252
0x000a  0xe50f4188  0xe3a06000  0xe50f7190  0xe50f4194
0x000b  0xe50f5158  0xe7d54006  0xe3540072  0x0a00000a
0x000c  0xe3540067  0x0a000008  0xe3540062  0x0a000006
0x000d  0xe3540079  0x0a000004  0xe3540070  0x0a000002
0x000e  0xe3540063  0x0a000000  0xeaffffeb  0xe2866001
0x000f  0xe3560004  0xbaffffee  0xe7d54006  0xe3540000
0x0010  0x1affffe5  0xe8bd00f0  0xe1a0f00e  0x6e45000a
0x0011  0x20726574  0x65646f63  0x6b616d20  0x6e207265
0x0012  0x3a656d61  0x45000a20  0x7265746e  0x646f6320
0x0013  0x72622065  0x656b6165  0x616e2072  0x203a656d
0x0014  0x6e45000a  0x20726574  0x6978616d  0x206d756d
0x0015  0x626d756e  0x6f207265  0x75712066  0x65697265
0x0016  0x6c612073  0x65776f6c  0x0a203a64  0x646f4300
0x0017  0x65726265  0x72656b61  0x20736920  0x646f4300
0x0018  0x6b616d65  0x69207265  0x4d002073  0x6d697861
0x0019  0x6e206d75  0x65626d75  0x666f2072  0x65756720
0x001a  0x73657373  0x4800203a  0x00006961  0x00000000
0x001b  0x00000000  0x00000000  0x00000000  0x00000000
0x001c  0x00000000  0x00000000  0x00000000  0x00000000
0x001d  0x00000000  0x00000000  0x00000000  0x00000000
0x001e  0x00000000  0x48000000  0x006e6569  0x00000000
0x001f  0x00000000  0x00000000  0x00000000  0x00000000
```

Step 4: Query code entry

- We print on the console the message, “Code breaker name, this is guess number: R9”, where is the name entered in Stage 1, and R9 is the current number of guesses left. Then increment R9.
- The we use the ‘getcode’ function above with message “Code breaker name, please enter a 4-character secret code”
- Stores the entered query code in an array called ‘querycode’

- Redo it until R9 = number of query allowed



```
        bl newline
printMaxGuessAllowed:
        mov r4, #maxQueriesNoti
        str r4, .WriteString
        str r7, .WriteUnsignedNum
        bl newline
        bl newline
getSecretCode:
        push {r0, r1, r2}
        mov r0, #secretcode
        mov r1, #getCodeMss
        mov r2, #makerName
        bl getcode
        pop {r0, r1, r2}
        bl newline
        ldrb r7, maxGuessAllowed
        add r7, r7, #1
        mov r9, #1
breakerGuess:
        mov r6, #breakerName
        str r6, .WriteString
        mov r4, #guessMss
        str r4, .WriteString
        str r9, .WriteUnsignedNum
        bl newline
        push {r0, r1, r2}
        mov r0, #querycode
        mov r1, #getCodeMss
        mov r2, #breakerName
        bl getcode
        pop {r0, r1, r2}
        add r9, r9, #1
        cmp r9, r7
        bl newline
        blt breakerGuess
exit:
```

```
PC   0x00000114
LR   0x000000bc
SP   0x000fffe4
R12  0x00000000
R11  0x00000000
R10  0x00000000
R9   0x00000004
R8   0x00000000
R7   0x00000247
R6   0x00000000
R5   0x000002fc
R4   0x00000288
R3   0x00000000
R2   0x00000247
R1   0x00000288
R0   0x000002fc

Count 329
Status bits NZCV 1000
```

```
Hai, this is guess number: 4
Hai, please enter a 4-character secret
code
```

```
rrgb
```

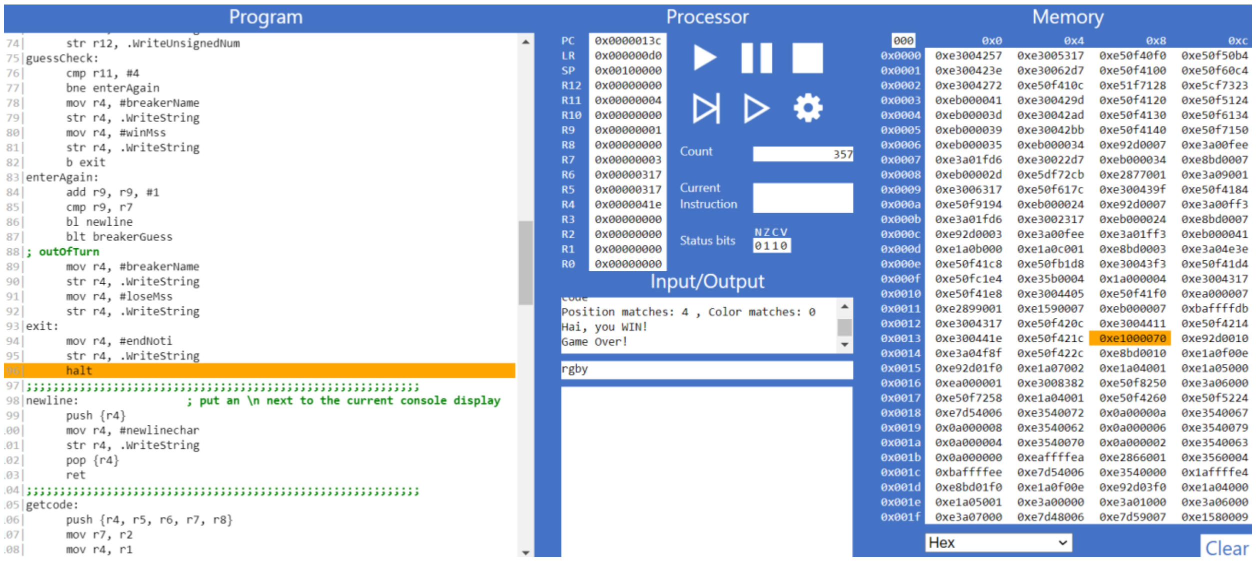
000	0x0	0x4	0x8	0xc
0x0000	0xe3004187	0xe3005247	0xe50f40f0	0xe50f50b4
0x0001	0xe300416e	0xe3006207	0xe50f4100	0xe50f60c4
0x0002	0xe30041a2	0xe50f410c	0xe51f7128	0xe5cf7253
0x0003	0xeb000027	0xe30041cd	0xe50f4120	0xe50f5124
0x0004	0xeb000023	0xe30041dd	0xe50f4130	0xe50f6134
0x0005	0xeb00001f	0xe30041eb	0xe50f4140	0xe50f7150
0x0006	0xeb00001b	0xeb00001a	0xe92d0007	0xe3a00fba
0x0007	0xe3a01fa2	0xe3002207	0xeb00001a	0xe8bd0007
0x0008	0xeb000013	0xe5df71fb	0xe2877001	0xe3a09001
0x0009	0xe3006247	0xe50f617c	0xe30042cf	0xe50f4184
0x000a	0xe50f9194	0xeb00000a	0xe92d0007	0xe3a00fbf
0x000b	0xe3a01fa2	0xe3002247	0xeb00000a	0xe8bd0007
0x000c	0xe2899001	0xe1590007	0xeb000001	0xbaffffef
0x000d	0xe1000070	0xe92d0010	0xe3a04f5b	0xe50f41c4
0x000e	0xe8bd0010	0xe1a0f00e	0xe92d00f0	0xe1a07002
0x000f	0xe1a04001	0xe1a05000	0xea000001	0xe30042b2
0x0010	0xe50f41e8	0xe3a06000	0xe50f71f0	0xe50f41f4
0x0011	0xe50f51b8	0xe7d54006	0xe3540072	0x0a00000a
0x0012	0xe3540067	0x0a000008	0xe3540062	0x0a000006
0x0013	0xe3540079	0x0a000004	0xe3540070	0x0a000002
0x0014	0xe3540063	0x0a000000	0xeaffffeb	0xe2866001
0x0015	0xe3560004	0xbaffffee	0xe7d54006	0xe3540000
0x0016	0x1affffe5	0xe8bd00f0	0xe1a0f00e	0x6e45000a
0x0017	0x20726574	0x65646f63	0x6b616d20	0x6e207265
0x0018	0x3a656d61	0x45000a20	0x7265746e	0x646f6320
0x0019	0x72622065	0x656b6165	0x616e2072	0x203a656d
0x001a	0x6e45000a	0x20726574	0x6978616d	0x206d756d
0x001b	0x626d756e	0x6f207265	0x75712066	0x65697265
0x001c	0x6c612073	0x65776f6c	0x0a203a64	0x646f4300
0x001d	0x65726265	0x72656b61	0x20736920	0x646f4300
0x001e	0x6b616d65	0x69207265	0x4d002073	0x6d697861
0x001f	0x6e206d75	0x65626d75	0x666f2072	0x65756720

Hex

Clear

Step 5a: Query code evaluation

- We create a 'comparecode' function that takes 2 parameters and return 2 outputs
- Accept both string codes as input parameters
- Count how many query pegs are in the correct position (Case 1)
- Count how many pegs are not in the same position but do have a colour match with at least one other peg in the secret code (Case 2).
- Both values should be passed back from the function as return values using R0 (Case 1 count), and R1 (Case 2 count)



```
        str r12, .WriteUnsignedNum
guessCheck:
        cmp r11, #4
        bne enterAgain
        mov r4, #breakerName
        str r4, .WriteString
        mov r4, #winMss
        str r4, .WriteString
        b exit
enterAgain:
        add r9, r9, #1
        cmp r9, r7
        bl newline
        blt breakerGuess
; outOfTurn
        mov r4, #breakerName
        str r4, .WriteString
        mov r4, #loseMss
        str r4, .WriteString
exit:
        mov r4, #endNoti
        str r4, .WriteString
        halt
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
newline:                ; put an \n next to the current console display
        push {r4}
        mov r4, #newlinechar
        str r4, .WriteString
        pop {r4}
        ret
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
getcode:
        push {r4, r5, r6, r7, r8}
        mov r7, r2
        mov r4, r1
```

```
PC   0x0000013c
LR   0x000000d0
SP   0x00100000
R12  0x00000000
R11  0x00000004
R10  0x00000000
R9   0x00000001
R8   0x00000000
R7   0x00000003
R6   0x00000317
R5   0x00000317
R4   0x0000041e
R3   0x00000000
R2   0x00000000
R1   0x00000000
R0   0x00000000

Count 357
Status bits NZCV 0110
```

```
code
Position matches: 4 , Color matches: 0
Hai, you WIN!
Game Over!
```

```
rgby
```

000	0x0	0x4	0x8	0xc
0x0000	0xe3004257	0xe3005317	0xe50f40f0	0xe50f50b4
0x0001	0xe300423e	0xe30062d7	0xe50f4100	0xe50f60c4
0x0002	0xe3004272	0xe50f410c	0xe51f7128	0xe5cf7323
0x0003	0xeb000041	0xe300429d	0xe50f4120	0xe50f5124
0x0004	0xeb00003d	0xe30042ad	0xe50f4130	0xe50f6134
0x0005	0xeb000039	0xe30042bb	0xe50f4140	0xe50f7150
0x0006	0xeb000035	0xeb000034	0xe92d0007	0xe3a00fee
0x0007	0xe3a01fd6	0xe30022d7	0xeb000034	0xe8bd0007
0x0008	0xeb00002d	0xe5df72cb	0xe2877001	0xe3a09001
0x0009	0xe3006317	0xe50f617c	0xe300439f	0xe50f4184
0x000a	0xe50f9194	0xeb000024	0xe92d0007	0xe3a00ff3
0x000b	0xe3a01fd6	0xe3002317	0xeb000024	0xe8bd0007
0x000c	0xe92d0003	0xe3a00fee	0xe3a01ff3	0xeb000041
0x000d	0xe1a0b000	0xe1a0c001	0xe8bd0003	0xe3a04e3e
0x000e	0xe50f41c8	0xe50fb1d8	0xe30043f3	0xe50f41d4
0x000f	0xe50fc1e4	0xe35b0004	0x1a000004	0xe3004317
0x0010	0xe50f41e8	0xe3004405	0xe50f41f0	0xea000007
0x0011	0xe2899001	0xe1590007	0xeb000007	0xbaffffdb
0x0012	0xe3004317	0xe50f420c	0xe3004411	0xe50f4214
0x0013	0xe300441e	0xe50f421c	0xe1000070	0xe92d0010
0x0014	0xe3a04f8f	0xe50f422c	0xe8bd0010	0xe1a0f00e
0x0015	0xe92d01f0	0xe1a07002	0xe1a04001	0xe1a05000
0x0016	0xea000001	0xe3008382	0xe50f8250	0xe3a06000
0x0017	0xe50f7258	0xe1a04001	0xe50f4260	0xe50f5224
0x0018	0xe7d54006	0xe3540072	0x0a00000a	0xe3540067
0x0019	0x0a000008	0xe3540062	0x0a000006	0xe3540079
0x001a	0x0a000004	0xe3540070	0x0a000002	0xe3540063
0x001b	0x0a000000	0xeaffffea	0xe2866001	0xe3560004
0x001c	0xbaffffee	0xe7d54006	0xe3540000	0x1affffe4
0x001d	0xe8bd01f0	0xe1a0f00e	0xe92d03f0	0xe1a04000
0x001e	0xe1a05001	0xe3a00000	0xe3a01000	0xe3a06000
0x001f	0xe3a07000	0xe7d48006	0xe7d59007	0xe1580009

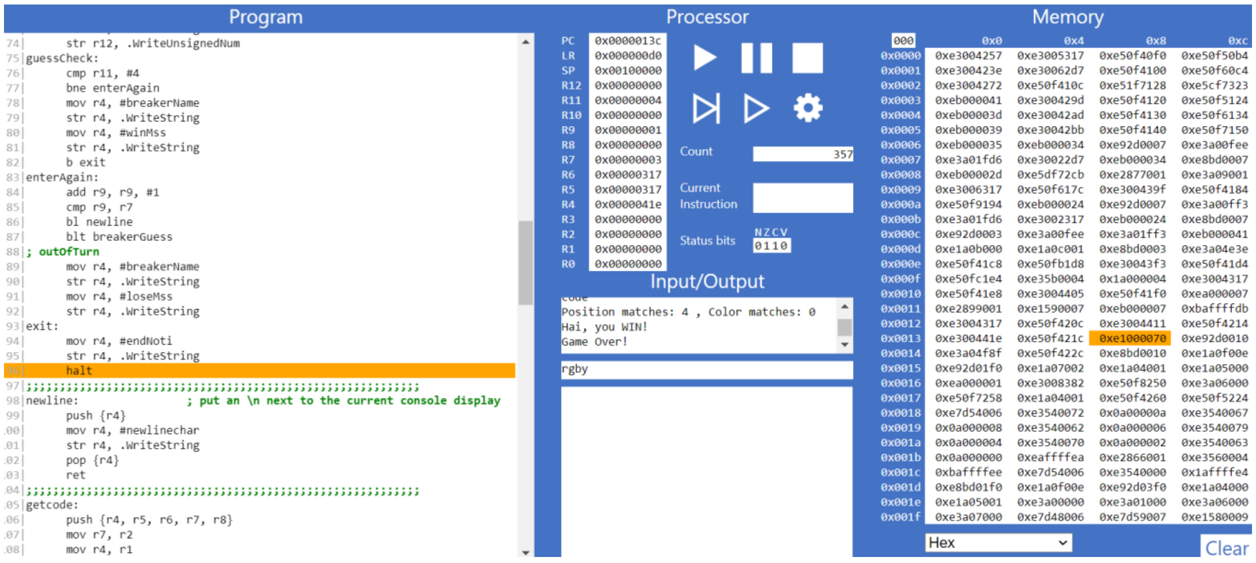
Hex

Clear

Step 5b: Query code evaluation

- After each iteration, print the value returned from the 'comparecode' function
- If R0 (number of position matches) = 4 then print "Code breaker name, you WIN", if R9 = 0, print "Code breaker name, you LOSE"

- Eventually, print on console “game over”



Program

```
        str r12, .WriteUnsignedNum
guessCheck:
        cmp r11, #4
        bne enterAgain
        mov r4, #breakerName
        str r4, .WriteString
        mov r4, #winMss
        str r4, .WriteString
        b exit
enterAgain:
        add r9, r9, #1
        cmp r9, r7
        bl newline
        blt breakerGuess
; outOfTurn
        mov r4, #breakerName
        str r4, .WriteString
        mov r4, #loseMss
        str r4, .WriteString
exit:
        mov r4, #endNoti
        str r4, .WriteString
        halt
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
newline:                    ; put an \n next to the current console display
        push {r4}
        mov r4, #newlinechar
        str r4, .WriteString
        pop {r4}
        ret
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
getcode:
        push {r4, r5, r6, r7, r8}
        mov r7, r2
        mov r4, r1
```

Processor

```
PC   0x0000013c
LR   0x000000d0
SP   0x00100000
R12  0x00000000
R11  0x00000004
R10  0x00000000
R9   0x00000001
R8   0x00000000
R7   0x00000003
R6   0x00000317
R5   0x00000317
R4   0x0000041e
R3   0x00000000
R2   0x00000000
R1   0x00000000
R0   0x00000000
```

Count 357

Current Instruction

Status bits NZCV 0110

Input/Output

```
Position matches: 4 , Color matches: 0
Hai, you WIN!
Game Over!
```

```
rgby
```

Memory

000	0x0	0x4	0x8	0xc
0x0000	0xe3004257	0xe3005317	0xe50f40f0	0xe50f50b4
0x0001	0xe300423e	0xe30062d7	0xe50f4100	0xe50f60c4
0x0002	0xe3004272	0xe50f410c	0xe51f7128	0xe5cf7323
0x0003	0xeb000041	0xe300429d	0xe50f4120	0xe50f5124
0x0004	0xeb00003d	0xe30042ad	0xe50f4130	0xe50f6134
0x0005	0xeb000039	0xe30042bb	0xe50f4140	0xe50f7150
0x0006	0xeb000035	0xeb000034	0xe92d0007	0xe3a00fee
0x0007	0xe3a01fd6	0xe30022d7	0xeb000034	0xe8bd0007
0x0008	0xeb00002d	0xe5df72cb	0xe2877001	0xe3a09001
0x0009	0xe3006317	0xe50f617c	0xe300439f	0xe50f4184
0x000a	0xe50f9194	0xeb000024	0xe92d0007	0xe3a00ff3
0x000b	0xe3a01fd6	0xe3002317	0xeb000024	0xe8bd0007
0x000c	0xe92d0003	0xe3a00fee	0xe3a01ff3	0xeb000041
0x000d	0xe1a0b000	0xe1a0c001	0xe8bd0003	0xe3a04e3e
0x000e	0xe50f41c8	0xe50fb1d8	0xe30043f3	0xe50f41d4
0x000f	0xe50fc1e4	0xe35b0004	0x1a000004	0xe3004317
0x0010	0xe50f41e8	0xe3004405	0xe50f41f0	0xea000007
0x0011	0xe2899001	0xe1590007	0xeb000007	0xbaffffdb
0x0012	0xe3004317	0xe50f420c	0xe3004411	0xe50f4214
0x0013	0xe300441e	0xe50f421c	0xe1000070	0xe92d0010
0x0014	0xe3a04f8f	0xe50f422c	0xe8bd0010	0xe1a0f00e
0x0015	0xe92d01f0	0xe1a07002	0xe1a04001	0xe1a05000
0x0016	0xea000001	0xe3008382	0xe50f8250	0xe3a06000
0x0017	0xe50f7258	0xe1a04001	0xe50f4260	0xe50f5224
0x0018	0xe7d54006	0xe3540072	0x0a00000a	0xe3540067
0x0019	0x0a000008	0xe3540062	0x0a000006	0xe3540079
0x001a	0x0a000004	0xe3540070	0x0a000002	0xe3540063
0x001b	0x0a000000	0xeaffffea	0xe2866001	0xe3560004
0x001c	0xbaffffee	0xe7d54006	0xe3540000	0x1affffe4
0x001d	0xe8bd01f0	0xe1a0f00e	0xe92d03f0	0xe1a04000
0x001e	0xe1a05001	0xe3a00000	0xe3a01000	0xe3a06000
0x001f	0xe3a07000	0xe7d48006	0xe7d59007	0xe1580009