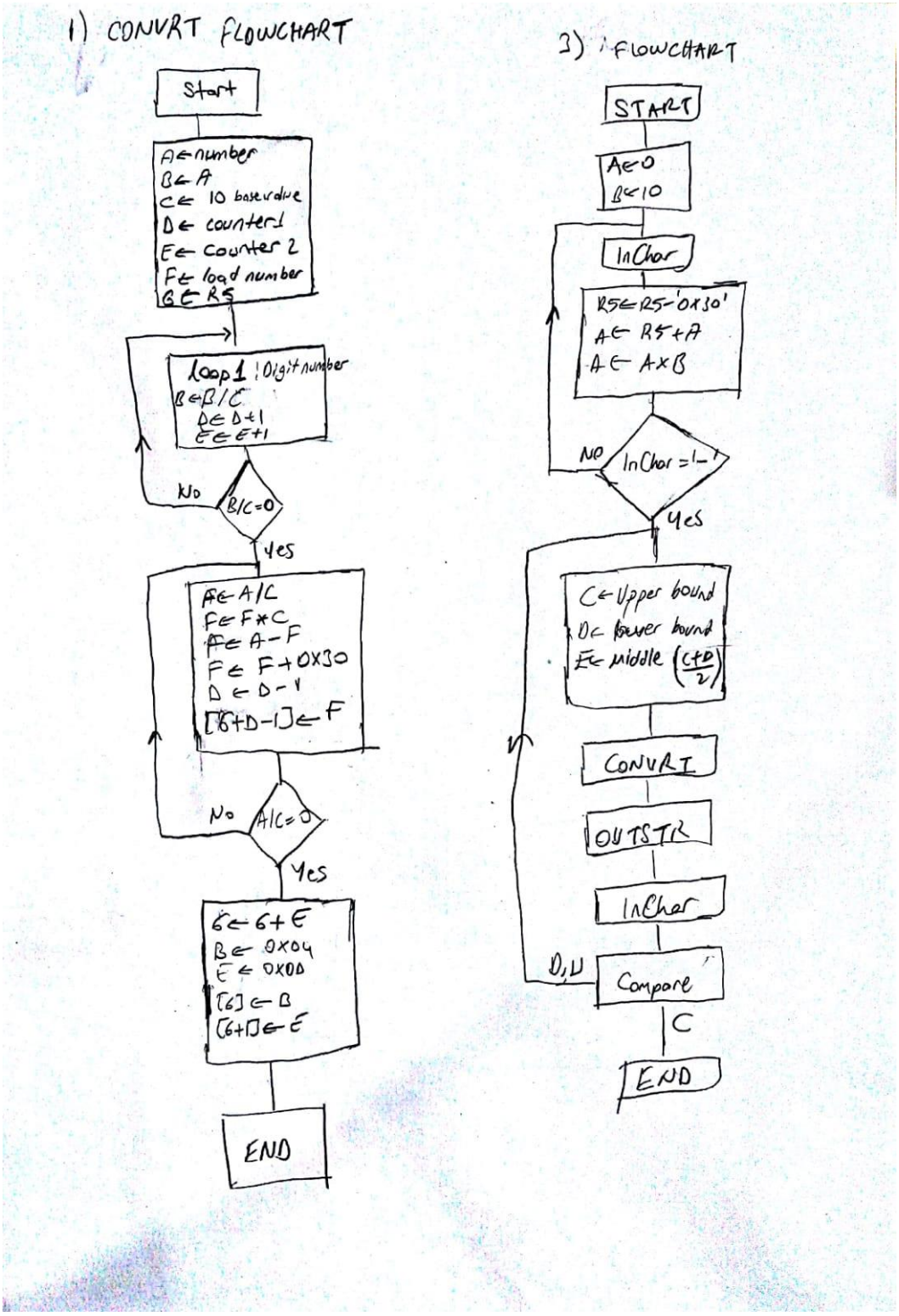
ALI AYDIN 2093326

EE447 PRELIMINARY REPORT 1



ALI AYDIN 2093326

1-) CONVRT.s

```
        AREA    routines, CODE, READONLY
        THUMB
        EXPORT CONVRT

CONVRT  PROC
        PUSH    {R0-R4}         ; preserve info in registers
        ;STR    R4, [R0]
        ;LDR    R7, [R4]        ;R6=R4

        MOV     R0, #10         ;base value
        MOV     R1, #0
        ADD     R7, R4, R1
        MOV     R2, #0
        MOV     R3, #0

loop1   ;loop1 to find, number how many digits have
        UDIV    R7, R7, R0      ;number is divided by 10 since to obtain next digit
        ADD     R2, #1          ;add 1 to counter1
        ADD     R3, #1          ;add 1 to counter2
        CMP     R7, #0x0        ;
        BEQ     loop2
        B       loop1

loop2   UDIV    R1, R4, R0      ;number is divided by 10
        MUL     R1, R0          ;number is multiplied by 10
        SUB     R1, R4, R1      ;I subtract R1 from R4 to find first digit of decimal number
        ADD     R1, #0x30       ;to convert ascii string constant, i added 0x30
        SUB     R2, #1
        STRB    R1, [R5, R2]    ;store digit in R5
        UDIV    R4, R4, R0      ;number is divided by 10 since to obtain next digit
        CMP     R4, #0x0
        BEQ     end_of_operation
        B       loop2

end_of_operation
        ADD     R5, R3
        MOV     R7, #0x04
        MOV     R3, #0x0D
        STRB    R3, [R5], #1    ; end of the transmission
        STRB    R7, [R5]        ;new linE
        POP     {R0-R4}         ;take info on stack
        BX      LR
        ENDP
        ALIGN
        END
```

2-)

ALI AYDIN 2093326

MAIN.s

```
1
2  NUM         EQU 0x20004000 ; NUM memory location
3  FIRST       EQU 0x20000400 ;address for storing digits
4  NUMBER      EQU 0x000ABC45 ;number will be converted
5
6              AREA    main, CODE, READONLY
7              THUMB
8              EXTERN CONVRT
9              IMPORT InChar
10             EXTERN OutStr
11             EXPORT __main
12
13 __main
14 loop        BL  InChar              ;determine to press any key
15             CMP R5, #00             ;compare any key
16             BEQ loop                ;if any key did not press, back to loop and wait for key
17             LDR R0, =NUM            ;load NUM to r0
18             LDR R1, =NUMBER         ;load number to r1
19             LDR R5, =FIRST          ;load address to r5
20
21             STR R1, [R0]            ;to store number in the address of NUM
22             LDR R4, [R0]            ;to load number in NUM to r4
23             BL  CONVRT              ;to convert number to decimal, go branch
24             LDR R5, =FIRST          ;load again address
25             BL  OutStr              ;to write number on termite
26             B   loop                ;back to loop
27
28
29             ALIGN
30             END
```

3-) UPBND.s

```
4          AREA    routines, CODE, READONLY
5          THUMB
6          EXPORT UPBND
7
8  UPBND   PROC
9          CMP     R5,#0x55        ;compare input U(up)
10         ADDEQ   R2,R4,#1        ;if U(up), add middle to 1 for new lower bound
11         CMP     R5,#0x44        ;compare input D(down)
12         SUBEQ   R3,R4,#1        ;if D(down), subtract middle to 1 for new upper bound
13         BX      LR              ;go to main
14
15         ENDP
16         ALIGN
17         END
```

MAIN.s

```
FIRST   EQU 0x20000400
        AREA    main, CODE, READONLY
        THUMB
        EXTERN CONVRT
        IMPORT InChar
        EXTERN OutStr
        IMPORT UPBND
        EXPORT __main
__main
start   MOV R0, #0          ;to store n value
        MOV R1, #10         ;to take 2 digit decimal number
        BL  InChar          ;take 2nd digit of the number
        SUB R5, #0x30       ;to take string as number, eliminate offset
        ADD R0, R5          ;add number to r0
        MUL R0, R1          ;multiply with 10 since 2nd digit of number
        BL  InChar          ;take 1st digit of the number
        SUB R5, #0x30       ;to take string as number, eliminate offset
        ADD R0, R5          ;add to 2nd digit
        LDR R2, =0x00       ;lower boundary
        LDR R3, =0x01       ;upper boundary
        LSL R3, R0          ;upper boundary = 2^n, shift r3 wrt n(input)
findingNumber
        ADD R4, R3, R2      ;= upper + lower boundaries
        LSR R4, #0x1        ;divide sum with 2 to obtain middle value
        LDR R5, =FIRST      ;load address to r5
        BL  CONVRT          ;convert number to decimal
        LDR R5, =FIRST      ;load address to r5
        BL  OutStr          ;write number to port
        BL  InChar          ;U(up) and D(down) or C(correct)
        CMP R5, #0x43       ;if correct, start beginning again
        BEQ start           ;go to start
        BL  UPBND           ;if not correct, determine new boundaries
        B   findingNumber   ;go to find number
        ALIGN
        END
```

4-) MAIN.s

```
FIRST           EQU     0x20000400
                AREA main,CODE,READONLY
                THUMB
                EXTERN InChar
                EXTERN OutStr
                EXTERN CONVRT
                EXTERN portals

                EXPORT __main
__main
start           MOV     R0,#0           ;input number
                MOV     R1,#10          ;base value

findingNumber   BL      InChar          ;take digit of input number
                CMP     R5,#0x2D        ;compare input with '-'
                BEQ     PORTAL          ;if equal, go to portal
                MUL     R0,R1           ;if not, multiply with current number with base value
                SUB     R5,#0x30        ;eliminate ascii offset
                ADD     R0,R5           ;add to number to storage register
                B       findingNumber   ;take next digit

PORTAL          MOV     R6,R0           ;keep number a register with no change
                BL      portals         ;go to portals subroutine
                MOV     R4,R0           ;load number to r4
                LDR     R5,=FIRST
                BL      CONVRT          ;convert to decimal
                LDR     R5,=FIRST
                BL      OutStr          ;show the decimal number
                B       start           ;go to start
                END
```

ALI AYDIN 2093326

Portals.s

```
 4                AREA subroutine,CODE,READONLY
 5                THUMB
 6                EXPORT portals
 7
 8  portals       PROC
 9                ;PUSH   {LR}
10                CMP     R0, #0          ;compare r0 to 0
11                BEQ     FINISH          ;if it is zero, go to finish
12                MOV     R4, #0          ;mode register
13                MOV     R2, #0
14                MOV     R7, #0
15                CMP     R0, #99         ;compare r0 to 99
16                BLS     START1          ;if less than 99, go to start1
17                ADD     R4, #1          ;if not, add mode register 1
18
19  START1
20                AND     R2, R0, #1      ;and r0 and 1, to find out whether number is odd or not
21                CMP     R2, #1
22                BEQ     START2          ;if not odd number, go to start2
23                ADD     R4, #4          ;if odd, add 2 to mode register
24                B       START3          ;and go to start3
25
26  START2
27                CMP     R0, #50         ;compare with 50
28                BLS     START3          ;if less than 50, go to start3
29                ADD     R4, #2          ;if not, add 4 to mode register
30
31  START3        MOV     R2, #7          ;r2 = 7
32                UDIV    R7, R0, R2      ;r7 = r0/7
33                MUL     R7, R2          ;r7 = r7*7
34                SUBS    R7, R0          ;r7 = r7 - r0 ==>result 0, then, number is multiple of 7
35                BNE     MAIN            ;if not, go main
36                ADD     R4, #8          ;if it is, add 8 to mode register
```

```
39  MAIN
40                ANDS    R7, R4, #8      ;
41                CMP     R7, #8          ;if mode register has 8
42                BEQ     PORTAL4         ;go to portal4
43                ANDS    R7, R4, #1
44                CMP     R7, #1          ;if mode register has 1
45                BEQ     PORTAL1         ;go to portal1
46                ANDS    R7, R4, #2
47                CMP     R7, #2          ;if mode register has 2
48                BEQ     PORTAL2         ;go to portal2
49                ANDS    R7, R4, #4
50                CMP     R7, #4          ;if mode register has 4
51                BEQ     PORTAL3         ;go to portal3
52                B       FINISH
53
54
55  PORTAL1       PUSH    {R0}
56                SUBS    R0, #47         ;r0 = r0 - 47
57                SUB     R4, #1          ;r4 = r4 - 1
58                PUSH    {R4}
59                PUSH    {LR}
60                B       portals         ;go to portals
61                POP     {LR}
62                POP     {R4}
63                POP     {R0}
64                B       MAIN
```

```
66  PORTAL2         PUSH    {R0}
67                  MOV     R3, #10         ;r3 = 10
68                  MOV     R2, #1          ;r2 = 1
69                  MOV     R7, #1
70  loop1           UDIV    R8, R0, R3      ;r8 = r0 / r3
71                  CMP     R0, #0          ;compare with 0
72                  BEQ     PORTAL2END      ;if 0, go to portal2end
73                  MLS     R9, R8, R3, R0  ;r9 = r0 - r8 * r3
74                  MOV     R0, R8          ;r0 = r8
75                  CMP     R9, #0
76                  BEQ     loop1           ;if r9 = 0, go to loop1
77                  MUL     R7, R9          ;if not, r7 = r7 * r9
78                  B       loop1           ;go to loop1
79  PORTAL2END      POP     {R0}
80                  SUB     R0,R7           ;r0 = r0 - r7
81                  SUB     R4,#2           ;r4 = r4 - 2
82                  PUSH    {R4}
83                  PUSH    {LR}
84                  B       portals
85                  POP     {LR}
86                  POP     {R4}
87                  POP     {R0}
88                  B       MAIN
89
90  PORTAL3         PUSH    {R0}
91                  LSR     R0, #1          ;r0 = r0 / 2
92                  SUB     R4, #4          ;r4 = r4 - 4
93                  PUSH    {R4}
94                  PUSH    {LR}
95                  B       portals
96                  POP     {LR}
97                  POP     {R4}
98                  POP     {R0}
99                  B       MAIN
```

```
101 PORTAL4
102                 PUSH    {R0}
103                 MOV     R2,#3           ;r2 = 3
104                 UDIV    R3, R0, R2      ;r3 = r0 / 3
105                 MUL     R3, R2          ;r3 = r3 * 3
106                 SUB     R0, R3          ;r0 = r0 - r3
107                 SUB     R4, #8          ;r4 = r4 - 8
108                 PUSH    {R4}
109                 PUSH    {LR}
110                 B       portals
111                 POP     {LR}
112                 POP     {R4}
113                 POP     {R0}
114                 B       MAIN
115
116 FINISH
117                 CMP     R6, R0          ;compare input number and result number
118                 BLS     BACK            ;if r6 < r0, go to back
119                 MOV     R6, R0          ;r6 = r0
120
121 BACK            ;POP        {LR}
122                 BX      LR
123
124
125                 ENDP
126                 END
```