**If/Else block to assembly:**

If P0.1 is 1:

Set R0.0 to 1

Else:

If R0.0 is 1:

Increment

Do Stuff

Set R0.0 to 0

;---------------

;Increment

;---------------

CheckIncBtn: JNB P0.1, MaybeInc ; Skip if button not pressed

SETB R0.0 ; Change state to pressed

SJMP CheckDecBtn ; Skip to next button

MaybeIncBtn: JNB R0.0, CheckDecBtn ; Skip if not pressed last check

LCALL IncMethod ; Increment counter

LCALL SetStateMethod ; Update LEDs and alarm duration

CLR R0.0 ; Change state to released

SJMP CheckDecBtn ; Skip to next button

**Ignore this. Just an attempt to make the if/else logic into a single method.**

;---------------

;CheckBtn

;---------------

CheckBtn: JNB P0.1, NotPressed

SETB R0.0 ; Change state to pressed

SJMP DoneChecking

NotPressed: JNB R0.0, DoneChecking

LCALL IncMethod ; Increment counter

LCALL SetStateMethod ; Update LEDs and alarm duration

CLR R0.0 ; Change state to released

DoneChecking: RET

MOV A,R3

JB PSW.6,OVER ;Checks if Aux flag is set, increment check

JB R3.7,OVER ;Checks if last bit of R3 is set, decrement check

OVER:

SETB P1.7 ;sets off beeper

LCALL DELAY ;calls time delay

;---------------

;SetStateMethod Sets the 4 LEDs to contents of R3’s first four bits

;---------------

ORG 300H

SetStateMethod:

MOV A,R3

MOV P2.4,ACC.3

MOV P2.7,ACC.2

MOV P2.5,ACC.1

MOV P2.6,ACC.0

RET

;---------------

;Sam’s (Nuke) Clears Everything

;---------------

MOV PSW,00h

MOV R3,00h

MOV R0,00h

MOV A,00h

MOV R1,00h

MOV R2,00h

*Alternative CheckForTurnOver*

CheckForTurnOver: MOV A, R3

MOV B, R0

MOV B.7, ACC.4

MOV R0, B

LCALL ClrTopAccNibble

MOV R3, A

TurnOverRET: RET

ClrTopAccNibble: CLR ACC.4 ; clear excess bits

CLR ACC.5

CLR ACC.6

CLR ACC.7

RET

*Delay (pseudo code)*

If R2 is not 0:

DEC R2

If R1 is not 0:

DEC R1