Abnormal Heartbeat Detector

Proyek Mikrokontroller 8051

Universitas Indonesia

Kelompok : Blue Pill

Ramadhan Kalih Sewu (1806148826)

Qisas Tazkia Hasanudin (1806200210)

Bryan Oliver (1806200305)

Farid Muhammad Rahman (1806148662)

FAKULTAS TEKNIK

DEPARTEMEN TEKNIK ELEKTRO

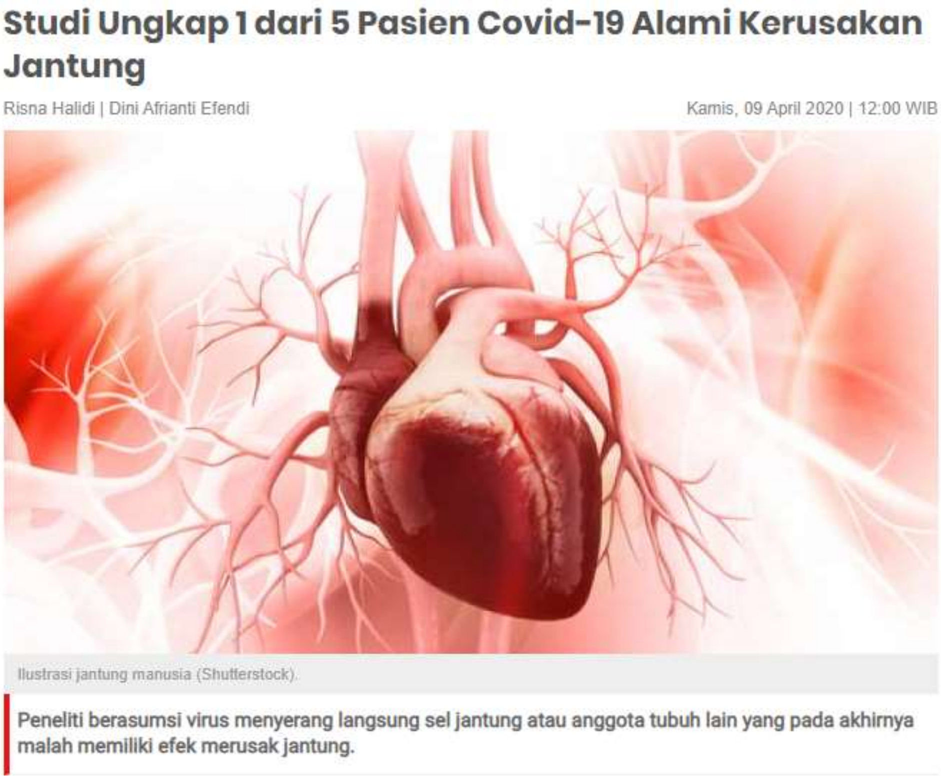
DEPOK 14 May 2020

I. Deskripsi sistem

Pemilihan Masalah :

Berdasarkan topik COVID-19, yang dapat berupa pencegahan, penanganan, dan pemantauan, kelompok kami mengambil satu topik yang berupa pemantauan. Hal ini didadasari oleh penemuan berita yang dilansirkan oleh Suara.com pada 9 April 2020. ‘Infeksi virus dapat menganggu aliran darah ke jantung, menyebabkan detak jantung tidak teratur dan gagal jantung.’ Dikutip dari Risna Halidi | Dini Afrianti Efendi https://www.suara.com/health/2020/04/09/120000/studi-ungkap-1-dari-5-pasien-covid-19-alami-kerusakan-jantung.

Gambar I: Pemicu Masalah



Studi Ungkap 1 dari 5 Pasien Covid-19 Alami Kerusakan Jantung

Risna Halidi | Dini Afrianti Efendi

Kamis, 09 April 2020 | 12:00 WIB

Ilustrasi jantung manusia (Shutterstock).

Peneliti berasumsi virus menyerang langsung sel jantung atau anggota tubuh lain yang pada akhirnya malah memiliki efek merusak jantung.

Tujuan Program :

Program ini dirancang untuk mendeteksi detak jantung dari sebuah sinyal square wave. Program akan memberi tahu kepada pengguna mengenai kondisi detak jantung berdasarkan umurnya. Tidak berhenti disitu, alat ini memiliki counter untuk menghitung berapa banyak penderita yang yang memiliki penyakit jantung. Sehingga dapat digunakan untuk statistik para tenaga medis dalam bekerja dilapangan dalam mengecek kondisi Jantung penderita Covid.

Sirkuit Embedded :

Rangkaian dibuat dengan prinsip simplicity, ease of use, dan interactive. Kami menggunakan seluruh port yang tersedia pada 8051 dengan upaya memanfaatkan seluruh Output yang ada sehingga data yang ditampilkan lengkap.

Input :

- Keypad
- Sensor (Square Wave Signal)

Output :

- LCD 16x2
- 2 BCD Seven Segment
- Serial Communication
- Buzzer
- Indikator LED

Seiring dengan jalannya rangkaian, akan dipandu oleh LCD 16x2 serta Serial Communication, pengguna tidak perlu khawatir dengan cara menggunakannya, karena setiap langkah telah dijelaskan secara perlahan.

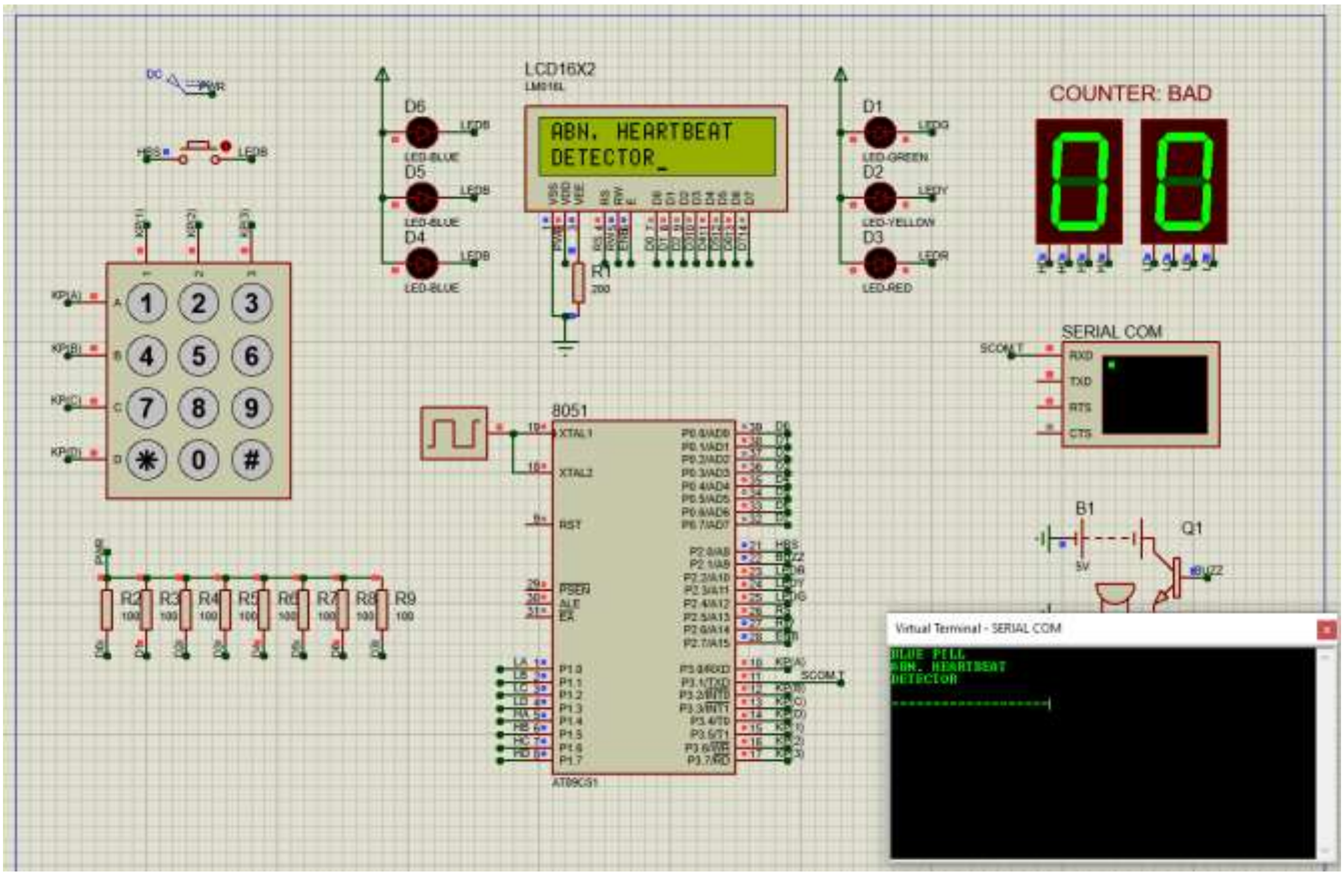
Cara Penggunaan Program :

Saat device baru dinyalakan, LCD dan serial monitor akan menampilkan boot-up text yang menunjukkan nama kelompok dan nama proyek. Lalu, program akan meminta user untuk memasukkan usianya melalui keypad, dengan usia minimal 00 dan maksimal 99. Setelah itu, program akan meminta user untuk meletakkan jarinya pada sensor, sambil menunggu sampai sensor mendeteksi perubahan cahaya, yang berasal dari tekanan darah pada jari user. Jika terdeteksi, device akan menghitung berapa banyak detak jantung yang terdeteksi oleh sensor selama 5 detik. Hasil ini akan dikalkulasi, sehingga didapatkan kecepatan detak jantung dalam satuan BPM.

Cara Kerja Program :

Untuk menentukan kualitas detak jantung user, telah diberikan parameter setiap umur yang memungkinkan pada Program yang di tulis di ROM. Program akan mengecek apabila Heartbeat diluar Lower Bound dan Upper Bound. Apabila kondisi tersebut terjadi maka Tampilkan BAD. Program mengambil angka Ideal, lalu mengecek -10 (Sebagai Lower Bound) dan +10 (Sebagai Upper Bound). Jika Heartbeat ada dalam range ideal, maka tampilkan GOOD. Kondisi lainnya tampilkan FAIR.

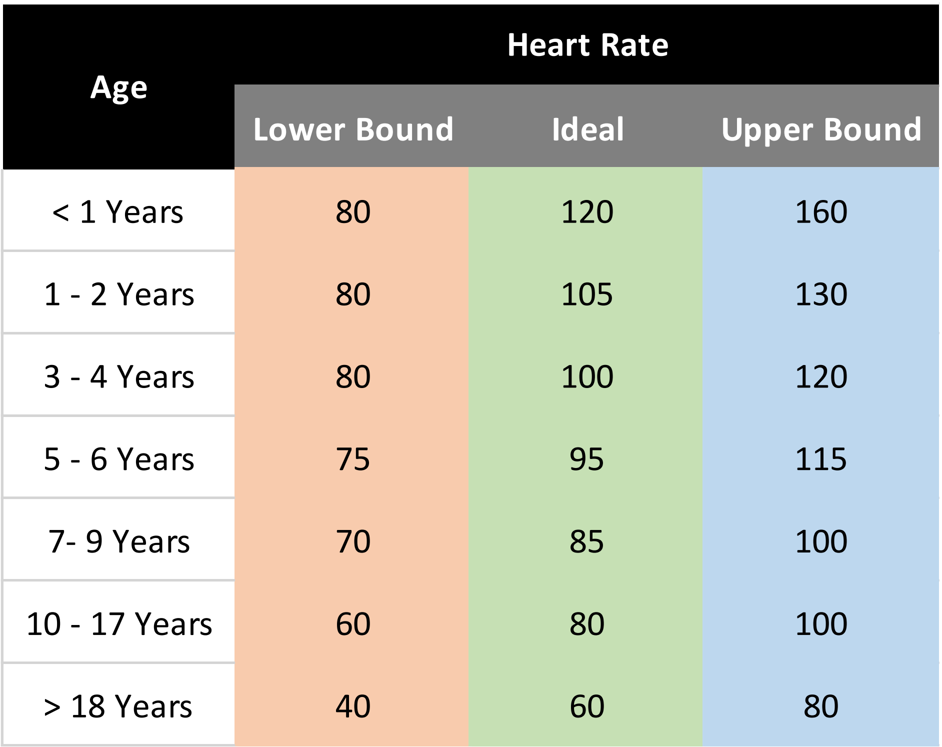
Gambar II : Rangkaian Sistem Embedded dalam Proteus



```
ABN. HEARTBEAT
DETECTOR_
```

```
BLUE PILL
ABN. HEARTBEAT
DETECTOR

-------------------------|
```

Gambar III : Data Referensi Heart Rate



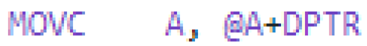
Age	Heart Rate		
	Lower Bound	Ideal	Upper Bound
< 1 Years	80	120	160
1 - 2 Years	80	105	130
3 - 4 Years	80	100	120
5 - 6 Years	75	95	115
7- 9 Years	70	85	100
10 - 17 Years	60	80	100
> 18 Years	40	60	80

Addressing Modes :

a) Register Addressing

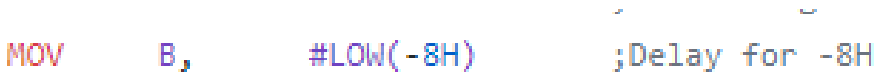
```
MOV      A, R6              ;MSB Saved in R6
```

b) Indexed Addressing



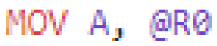
```
MOVC     A, @A+DPTR
```

c) Immediate Addressing



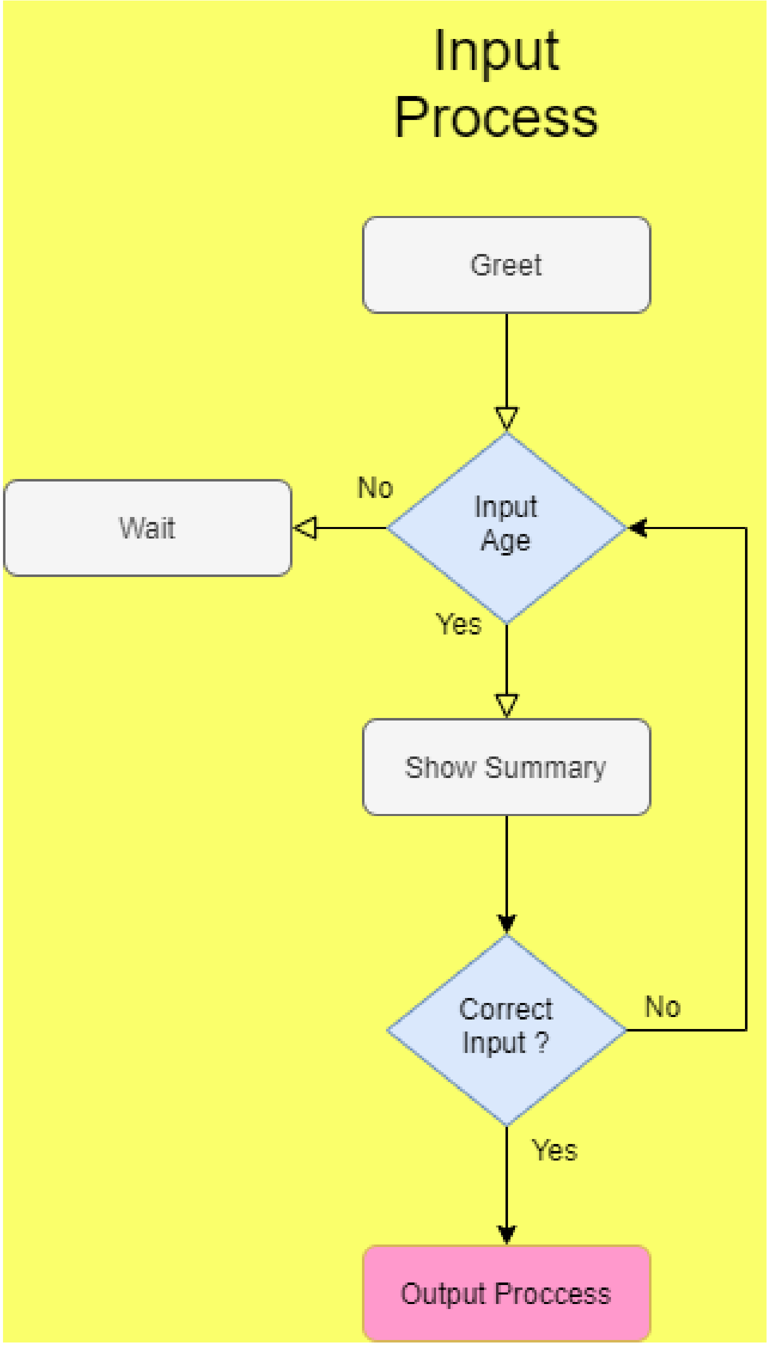
```
MOV      B,       #LOW(-8H)        ;Delay for -8H
```

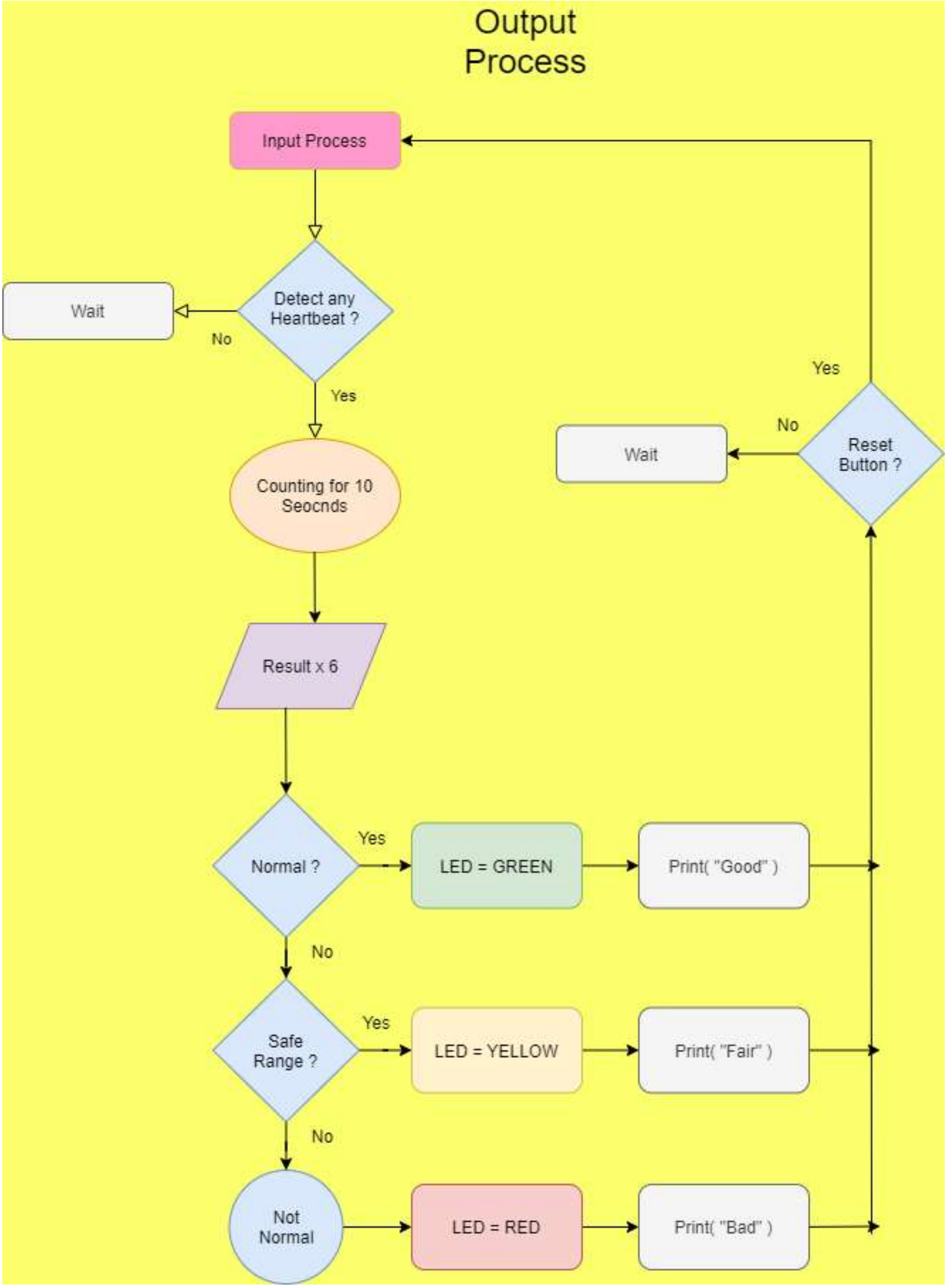
d) Register Indirect



```
MOV A, @R0
```

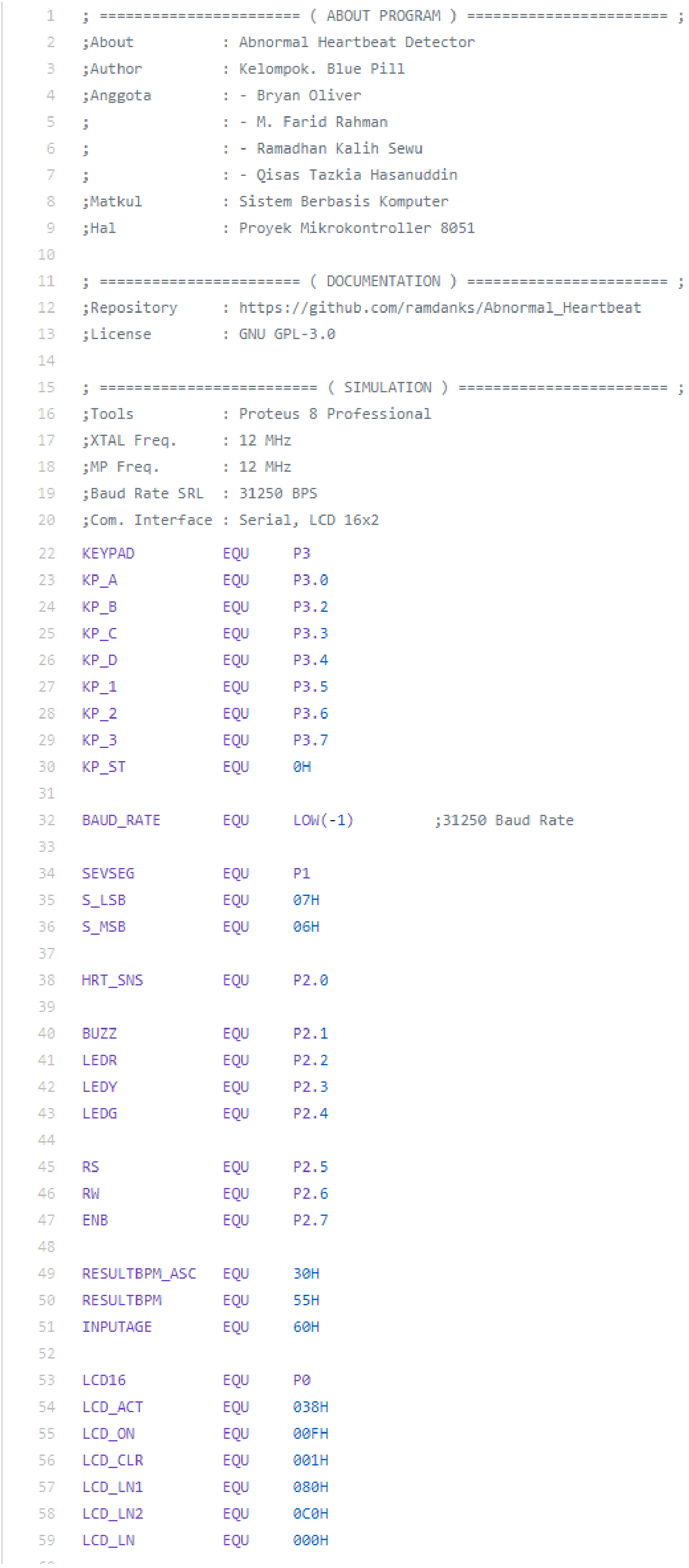
II. Flowchart sistem



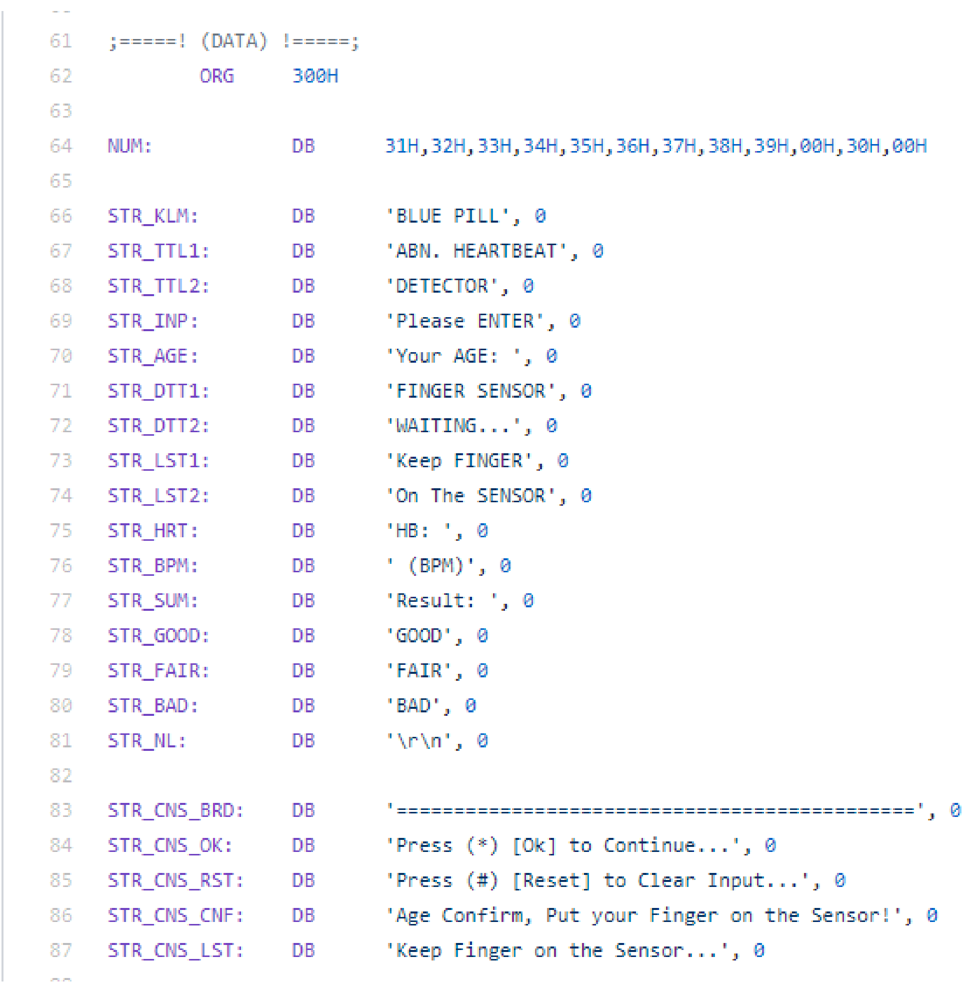
Output Process
Input Process
Detect any Heartbeat ?
No
Wait
Yes
Counting for 10 Seocnds
Result x 6
Normal ?
Yes
LED = GREEN
Print("Good")
No
Safe Range ?
Yes
LED = YELLOW
Print("Fair")
No
Not Normal
LED = RED
Print("Bad")
Reset Button ?
Yes
No
Wait

III. Source code

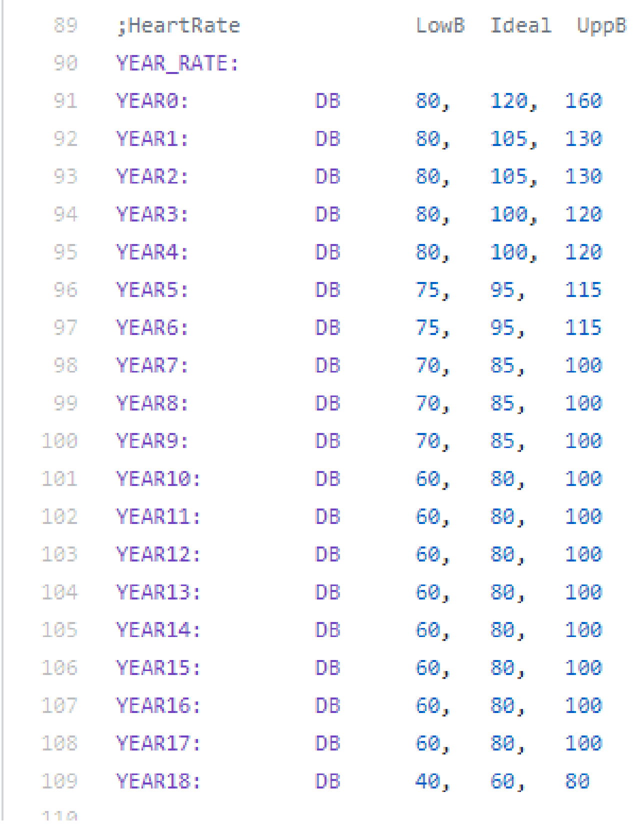
Repository : https://github.com/ramdanks/Abnormal_Heartbeat



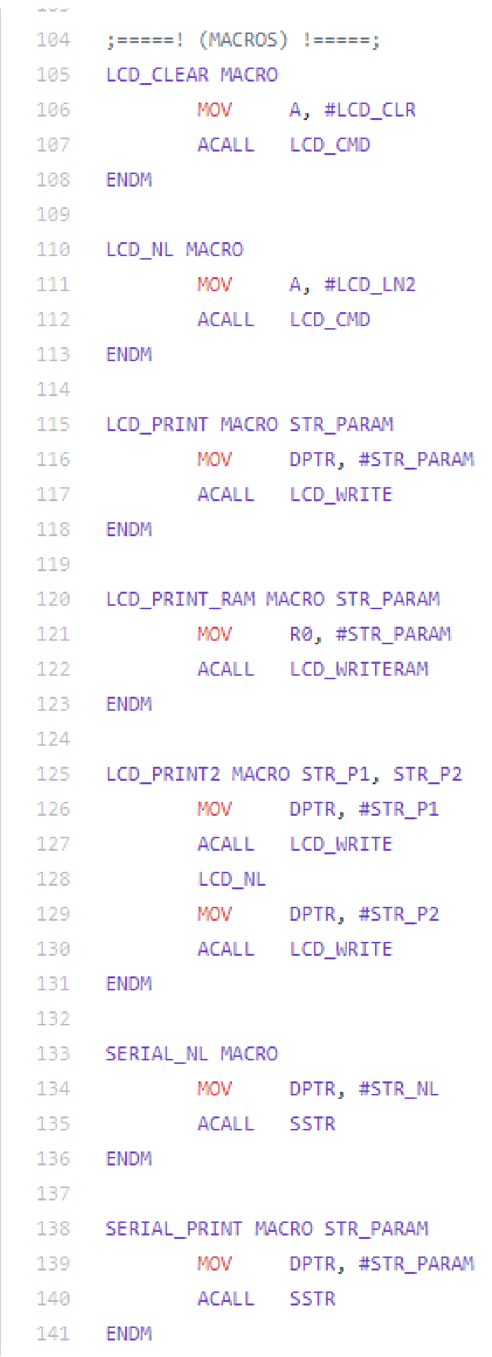
```
; ====================== ( ABOUT PROGRAM ) ====================== ;
;About          : Abnormal Heartbeat Detector
;Author         : Kelompok. Blue Pill
;Anggota        : - Bryan Oliver
;               : - M. Farid Rahman
;               : - Ramadhan Kalih Sewu
;               : - Qisas Tazkia Hasanuddin
;Matkul         : Sistem Berbasis Komputer
;Hal            : Proyek Mikrokontroller 8051

; ====================== ( DOCUMENTATION ) ====================== ;
;Repository     : https://github.com/ramdanks/Abnormal_Heartbeat
;License        : GNU GPL-3.0

; ======================== ( SIMULATION ) ======================== ;
;Tools          : Proteus 8 Professional
;XTAL Freq.     : 12 MHz
;MP Freq.       : 12 MHz
;Baud Rate SRL  : 31250 BPS
;Com. Interface : Serial, LCD 16x2
KEYPAD          EQU     P3
KP_A            EQU     P3.0
KP_B            EQU     P3.2
KP_C            EQU     P3.3
KP_D            EQU     P3.4
KP_1            EQU     P3.5
KP_2            EQU     P3.6
KP_3            EQU     P3.7
KP_ST           EQU     0H

BAUD_RATE       EQU     LOW(-1)         ;31250 Baud Rate

SEVSEG          EQU     P1
S_LSB           EQU     07H
S_MSB           EQU     06H

HRT_SNS         EQU     P2.0

BUZZ            EQU     P2.1
LEDR            EQU     P2.2
LEDY            EQU     P2.3
LEDG            EQU     P2.4

RS              EQU     P2.5
RW              EQU     P2.6
ENB             EQU     P2.7

RESULTBPM_ASC   EQU     30H
RESULTBPM       EQU     55H
INPUTAGE        EQU     60H

LCD16           EQU     P0
LCD_ACT         EQU     038H
LCD_ON          EQU     00FH
LCD_CLR         EQU     001H
LCD_LN1         EQU     080H
LCD_LN2         EQU     0C0H
LCD_LN          EQU     000H
```



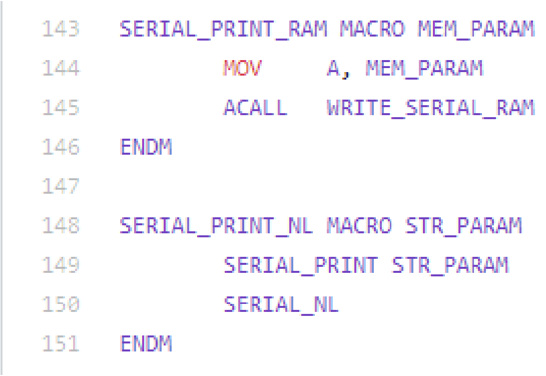
```
;=====! (DATA) !=====;
        ORG     300H

NUM:            DB      31H,32H,33H,34H,35H,36H,37H,38H,39H,00H,30H,00H

STR_KLM:        DB      'BLUE PILL', 0
STR_TTL1:       DB      'ABN. HEARTBEAT', 0
STR_TTL2:       DB      'DETECTOR', 0
STR_INP:        DB      'Please ENTER', 0
STR_AGE:        DB      'Your AGE: ', 0
STR_DTT1:       DB      'FINGER SENSOR', 0
STR_DTT2:       DB      'WAITING...', 0
STR_LST1:       DB      'Keep FINGER', 0
STR_LST2:       DB      'On The SENSOR', 0
STR_HRT:        DB      'HB: ', 0
STR_BPM:        DB      ' (BPM)', 0
STR_SUM:        DB      'Result: ', 0
STR_GOOD:       DB      'GOOD', 0
STR_FAIR:       DB      'FAIR', 0
STR_BAD:        DB      'BAD', 0
STR_NL:         DB      '\r\n', 0

STR_CNS_BRD:    DB      '===========================================', 0
STR_CNS_OK:     DB      'Press (*) [Ok] to Continue...', 0
STR_CNS_RST:    DB      'Press (#) [Reset] to Clear Input...', 0
STR_CNS_CNF:    DB      'Age Confirm, Put your Finger on the Sensor!', 0
STR_CNS_LST:    DB      'Keep Finger on the Sensor...', 0
```



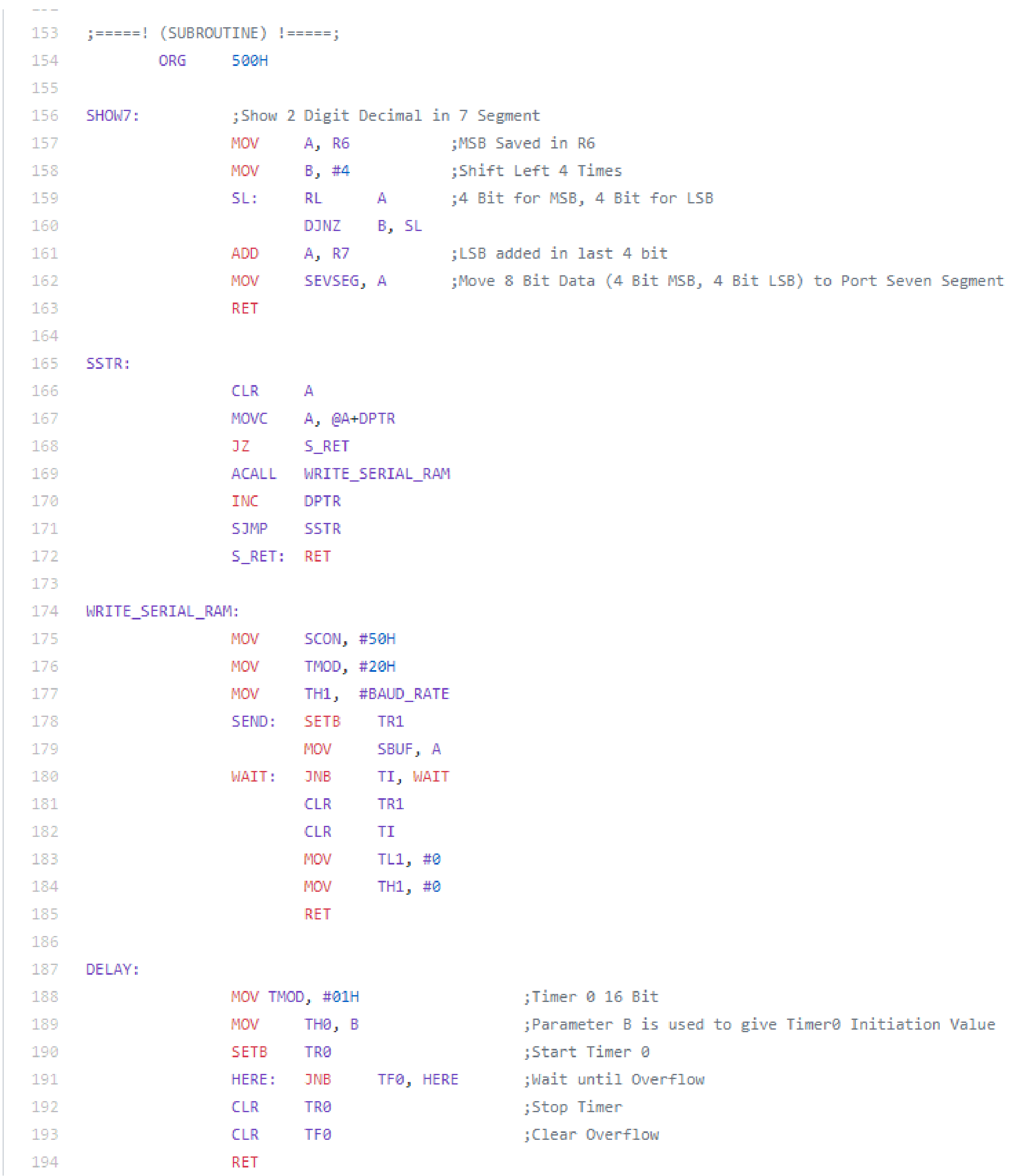
```
;HeartRate              LowB  Ideal  UppB
YEAR_RATE:
YEAR0:          DB      80,   120,  160
YEAR1:          DB      80,   105,  130
YEAR2:          DB      80,   105,  130
YEAR3:          DB      80,   100,  120
YEAR4:          DB      80,   100,  120
YEAR5:          DB      75,   95,   115
YEAR6:          DB      75,   95,   115
YEAR7:          DB      70,   85,   100
YEAR8:          DB      70,   85,   100
YEAR9:          DB      70,   85,   100
YEAR10:         DB      60,   80,   100
YEAR11:         DB      60,   80,   100
YEAR12:         DB      60,   80,   100
YEAR13:         DB      60,   80,   100
YEAR14:         DB      60,   80,   100
YEAR15:         DB      60,   80,   100
YEAR16:         DB      60,   80,   100
YEAR17:         DB      60,   80,   100
YEAR18:         DB      40,   60,   80
```



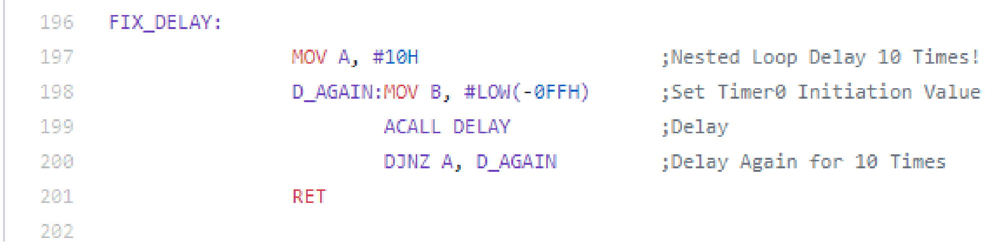
```
;=====! (MACROS) !=====;
LCD_CLEAR MACRO
        MOV     A, #LCD_CLR
        ACALL   LCD_CMD
ENDM

LCD_NL MACRO
        MOV     A, #LCD_LN2
        ACALL   LCD_CMD
ENDM

LCD_PRINT MACRO STR_PARAM
        MOV     DPTR, #STR_PARAM
        ACALL   LCD_WRITE
ENDM

LCD_PRINT_RAM MACRO STR_PARAM
        MOV     R0, #STR_PARAM
        ACALL   LCD_WRITERAM
ENDM

LCD_PRINT2 MACRO STR_P1, STR_P2
        MOV     DPTR, #STR_P1
        ACALL   LCD_WRITE
        LCD_NL
        MOV     DPTR, #STR_P2
        ACALL   LCD_WRITE
ENDM

SERIAL_NL MACRO
        MOV     DPTR, #STR_NL
        ACALL   SSTR
ENDM

SERIAL_PRINT MACRO STR_PARAM
        MOV     DPTR, #STR_PARAM
        ACALL   SSTR
ENDM
```



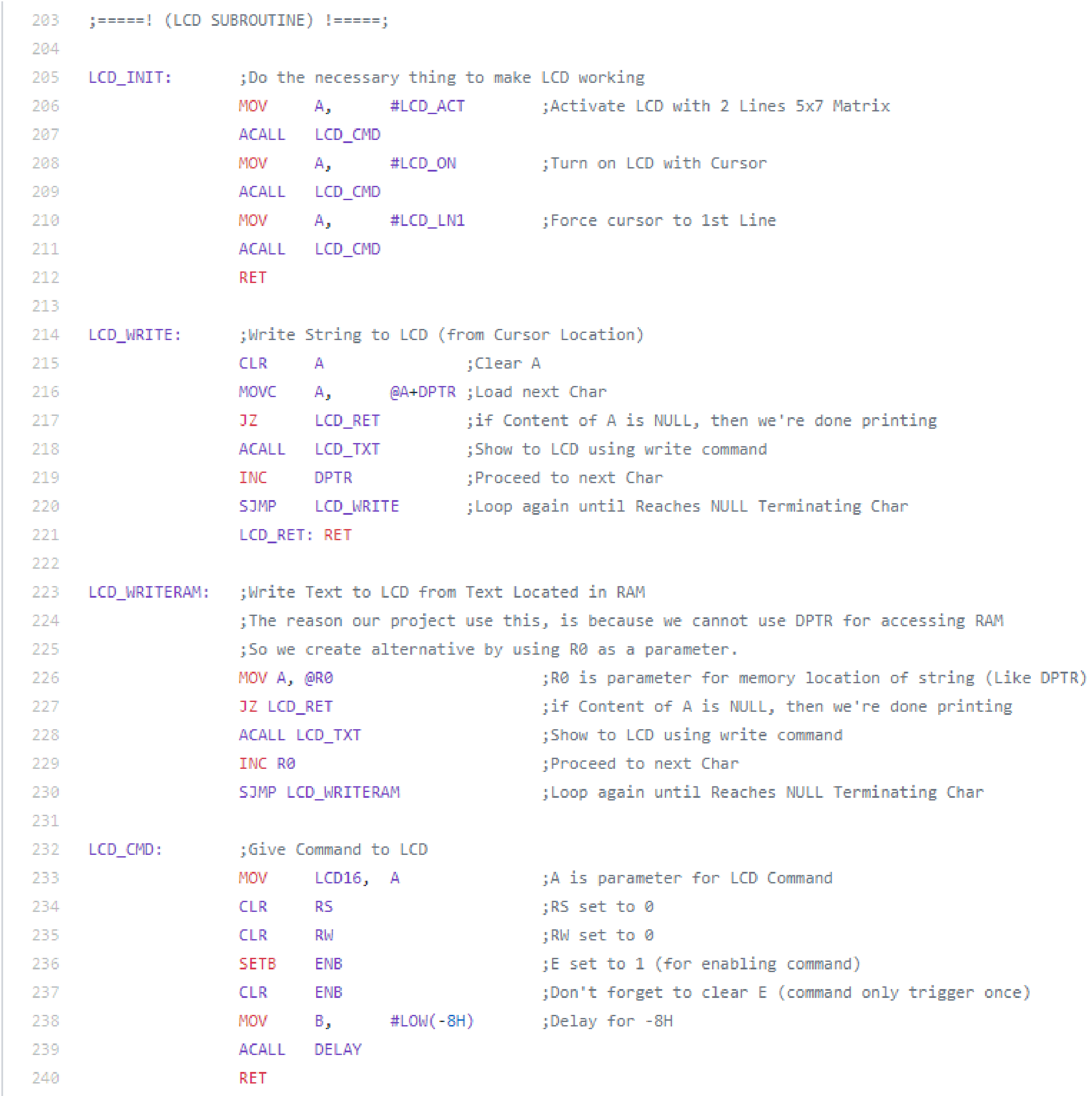
```
SERIAL_PRINT_RAM MACRO MEM_PARAM
        MOV     A, MEM_PARAM
        ACALL   WRITE_SERIAL_RAM
ENDM

SERIAL_PRINT_NL MACRO STR_PARAM
        SERIAL_PRINT STR_PARAM
        SERIAL_NL
ENDM
```



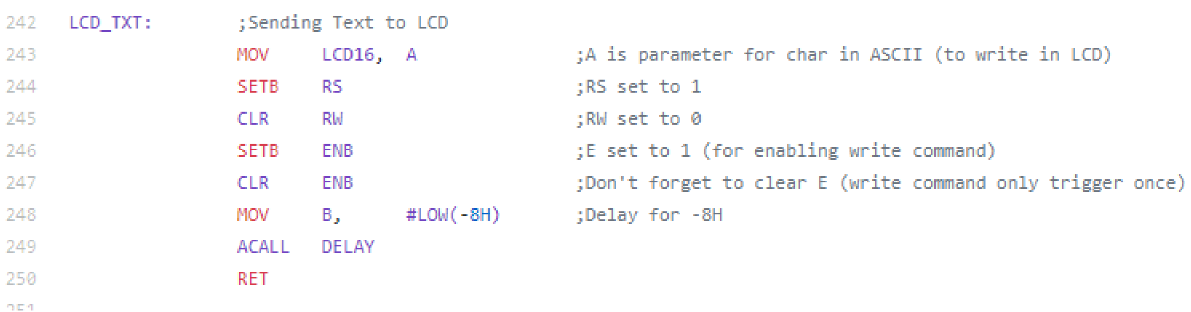
```
;=====! (SUBROUTINE) !=====;
        ORG     500H

SHOW7:          ;Show 2 Digit Decimal in 7 Segment
                MOV     A, R6           ;MSB Saved in R6
                MOV     B, #4           ;Shift Left 4 Times
                SL:     RL      A       ;4 Bit for MSB, 4 Bit for LSB
                        DJNZ    B, SL
                ADD     A, R7           ;LSB added in last 4 bit
                MOV     SEVSEG, A       ;Move 8 Bit Data (4 Bit MSB, 4 Bit LSB) to Port Seven Segment
                RET

SSTR:
                CLR     A
                MOVC    A, @A+DPTR
                JZ      S_RET
                ACALL   WRITE_SERIAL_RAM
                INC     DPTR
                SJMP    SSTR
                S_RET:  RET

WRITE_SERIAL_RAM:
                MOV     SCON, #50H
                MOV     TMOD, #20H
                MOV     TH1,  #BAUD_RATE
                SEND:   SETB    TR1
                        MOV     SBUF, A
                WAIT:   JNB     TI, WAIT
                        CLR     TR1
                        CLR     TI
                        MOV     TL1, #0
                        MOV     TH1, #0
                        RET

DELAY:
                MOV TMOD, #01H                  ;Timer 0 16 Bit
                MOV     TH0, B                  ;Parameter B is used to give Timer0 Initiation Value
                SETB    TR0                     ;Start Timer 0
                HERE:   JNB     TF0, HERE       ;Wait until Overflow
                CLR     TR0                     ;Stop Timer
                CLR     TF0                     ;Clear Overflow
                RET
```



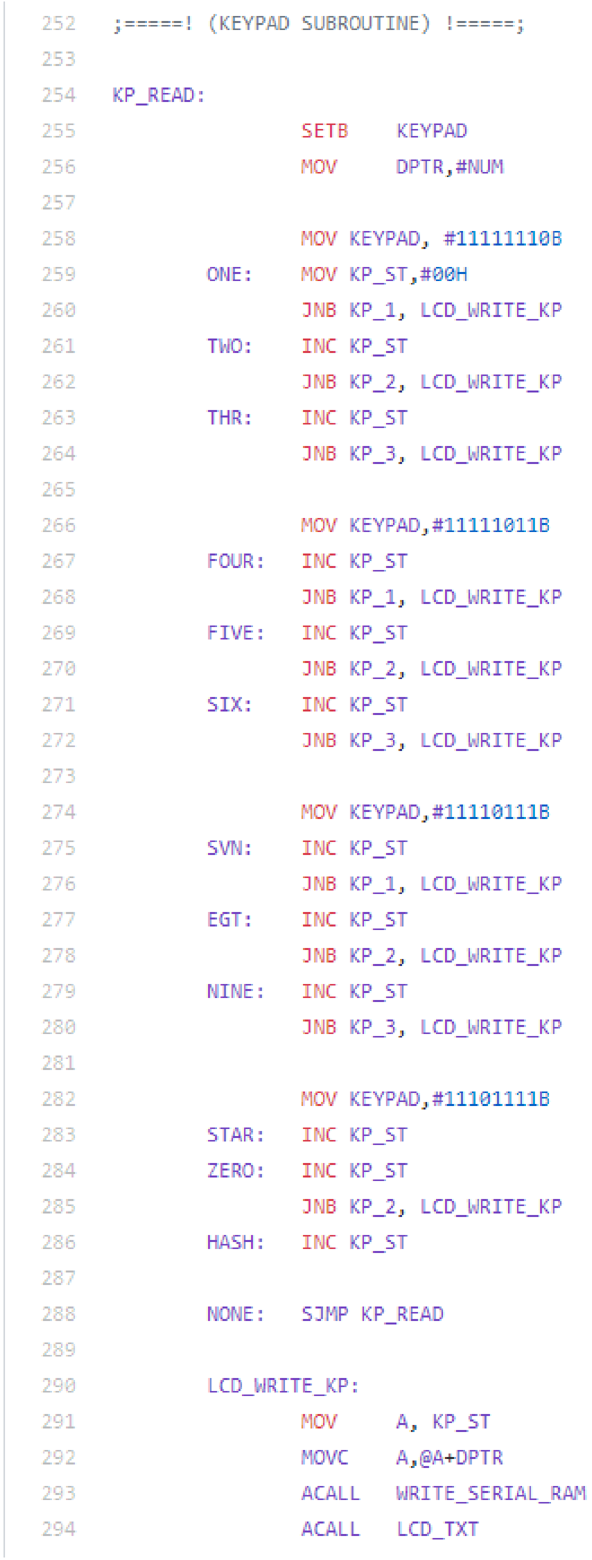
```
FIX_DELAY:
                MOV A, #10H                     ;Nested Loop Delay 10 Times!
                D_AGAIN:MOV B, #LOW(-0FFH)      ;Set Timer0 Initiation Value
                        ACALL DELAY             ;Delay
                        DJNZ A, D_AGAIN         ;Delay Again for 10 Times
                RET

```



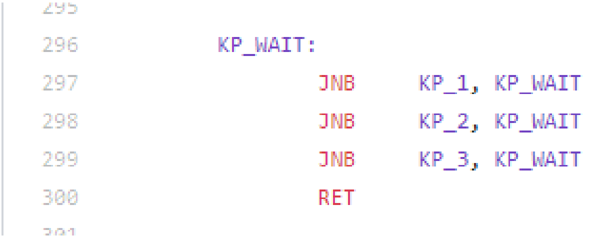
```
;=====! (LCD SUBROUTINE) !=====;

LCD_INIT:       ;Do the necessary thing to make LCD working
                MOV     A,      #LCD_ACT        ;Activate LCD with 2 Lines 5x7 Matrix
                ACALL   LCD_CMD
                MOV     A,      #LCD_ON         ;Turn on LCD with Cursor
                ACALL   LCD_CMD
                MOV     A,      #LCD_LN1        ;Force cursor to 1st Line
                ACALL   LCD_CMD
                RET

LCD_WRITE:      ;Write String to LCD (from Cursor Location)
                CLR     A               ;Clear A
                MOVC    A,      @A+DPTR ;Load next Char
                JZ      LCD_RET         ;if Content of A is NULL, then we're done printing
                ACALL   LCD_TXT         ;Show to LCD using write command
                INC     DPTR            ;Proceed to next Char
                SJMP    LCD_WRITE       ;Loop again until Reaches NULL Terminating Char
                LCD_RET: RET

LCD_WRITERAM:   ;Write Text to LCD from Text Located in RAM
                ;The reason our project use this, is because we cannot use DPTR for accessing RAM
                ;So we create alternative by using R0 as a parameter.
                MOV A, @R0                      ;R0 is parameter for memory location of string (Like DPTR)
                JZ LCD_RET                      ;if Content of A is NULL, then we're done printing
                ACALL LCD_TXT                   ;Show to LCD using write command
                INC R0                          ;Proceed to next Char
                SJMP LCD_WRITERAM               ;Loop again until Reaches NULL Terminating Char

LCD_CMD:        ;Give Command to LCD
                MOV     LCD16,  A               ;A is parameter for LCD Command
                CLR     RS                      ;RS set to 0
                CLR     RW                      ;RW set to 0
                SETB    ENB                     ;E set to 1 (for enabling command)
                CLR     ENB                     ;Don't forget to clear E (command only trigger once)
                MOV     B,      #LOW(-8H)       ;Delay for -8H
                ACALL   DELAY
                RET
```



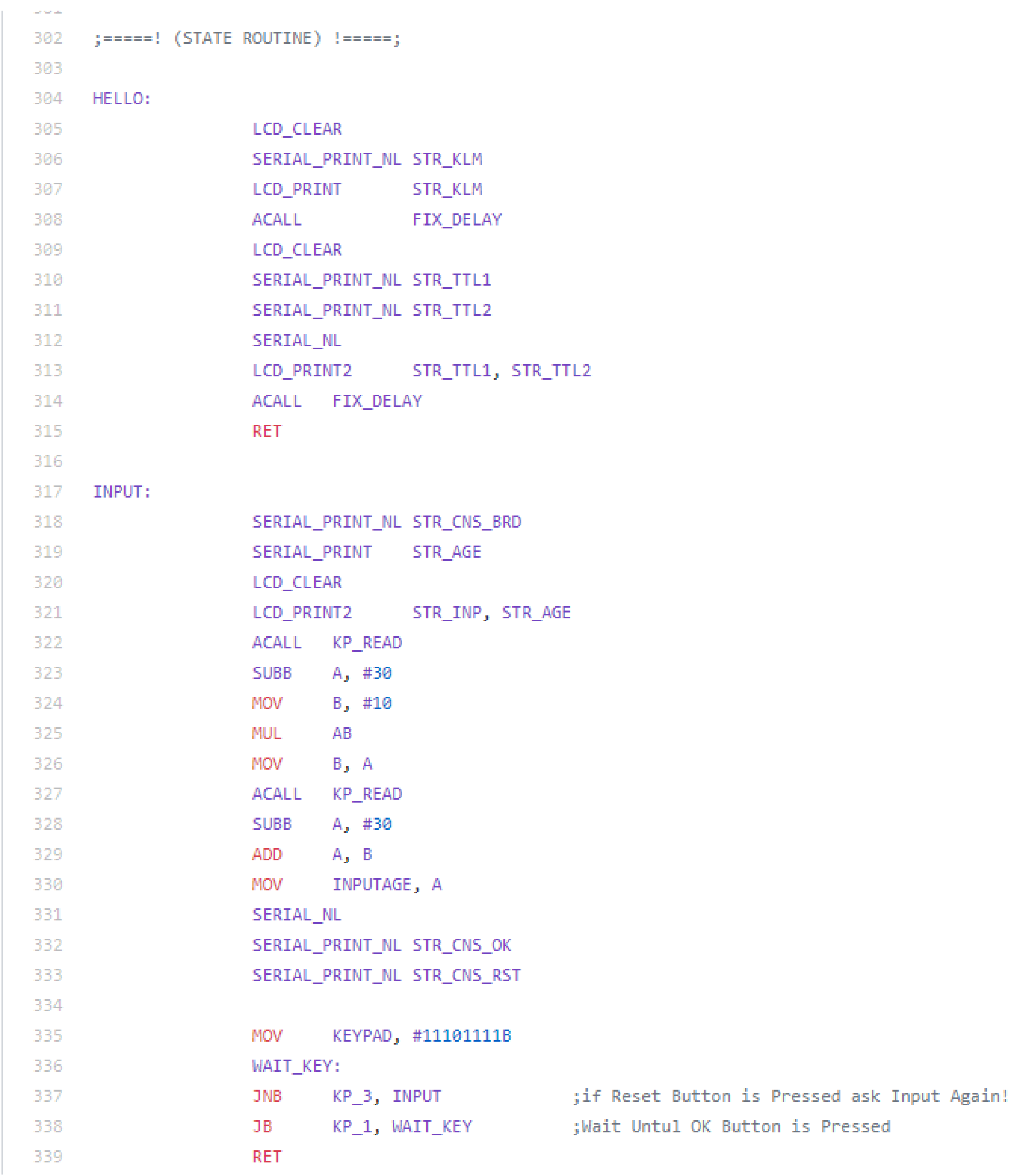
```
LCD_TXT:        ;Sending Text to LCD
                MOV     LCD16,  A               ;A is parameter for char in ASCII (to write in LCD)
                SETB    RS                      ;RS set to 1
                CLR     RW                      ;RW set to 0
                SETB    ENB                     ;E set to 1 (for enabling write command)
                CLR     ENB                     ;Don't forget to clear E (write command only trigger once)
                MOV     B,      #LOW(-8H)       ;Delay for -8H
                ACALL   DELAY
                RET
```



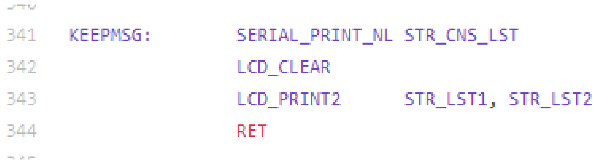
```
;=====! (KEYPAD SUBROUTINE) !=====;

KP_READ:
                SETB    KEYPAD
                MOV     DPTR,#NUM

                MOV KEYPAD, #11111110B
        ONE:    MOV KP_ST,#00H
                JNB KP_1, LCD_WRITE_KP
        TWO:    INC KP_ST
                JNB KP_2, LCD_WRITE_KP
        THR:    INC KP_ST
                JNB KP_3, LCD_WRITE_KP

                MOV KEYPAD,#11111011B
        FOUR:   INC KP_ST
                JNB KP_1, LCD_WRITE_KP
        FIVE:   INC KP_ST
                JNB KP_2, LCD_WRITE_KP
        SIX:    INC KP_ST
                JNB KP_3, LCD_WRITE_KP

                MOV KEYPAD,#11110111B
        SVN:    INC KP_ST
                JNB KP_1, LCD_WRITE_KP
        EGT:    INC KP_ST
                JNB KP_2, LCD_WRITE_KP
        NINE:   INC KP_ST
                JNB KP_3, LCD_WRITE_KP

                MOV KEYPAD,#11101111B
        STAR:   INC KP_ST
        ZERO:   INC KP_ST
                JNB KP_2, LCD_WRITE_KP
        HASH:   INC KP_ST

        NONE:   SJMP KP_READ

        LCD_WRITE_KP:
                MOV     A, KP_ST
                MOVC    A,@A+DPTR
                ACALL   WRITE_SERIAL_RAM
                ACALL   LCD_TXT
```



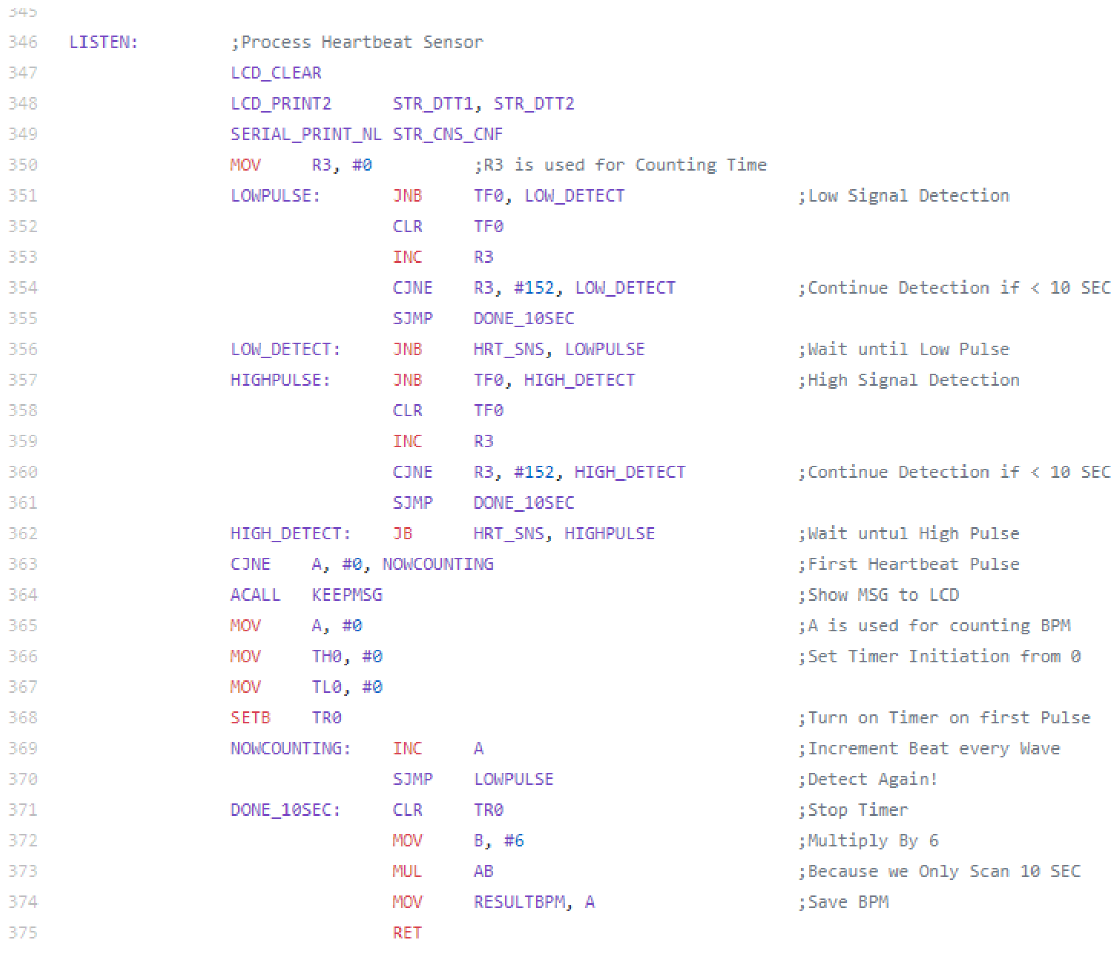
```
         KP_WAIT:
                 JNB     KP_1, KP_WAIT
                 JNB     KP_2, KP_WAIT
                 JNB     KP_3, KP_WAIT
                 RET
```



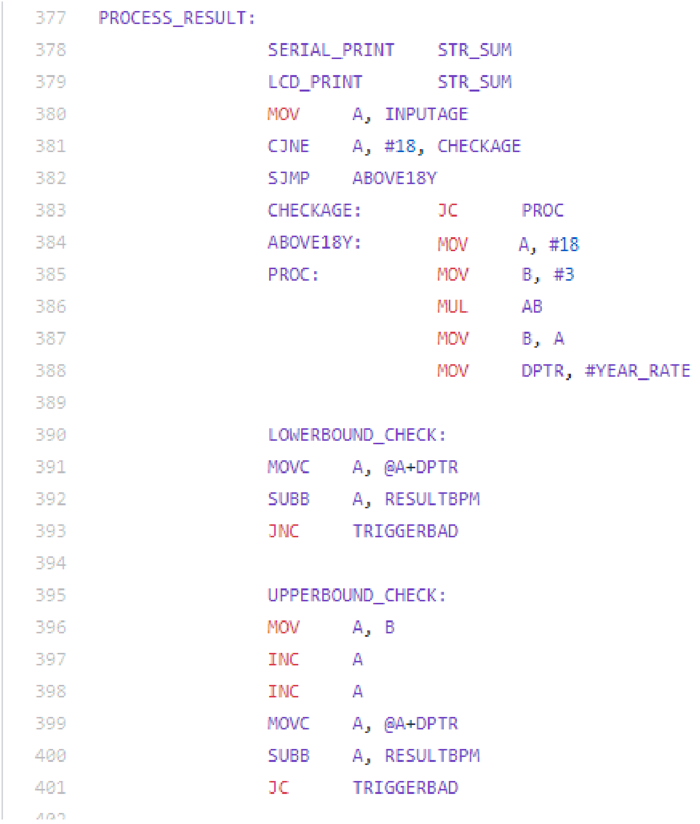
```
;=====! (STATE ROUTINE) !=====;

HELLO:
                LCD_CLEAR
                SERIAL_PRINT_NL STR_KLM
                LCD_PRINT       STR_KLM
                ACALL           FIX_DELAY
                LCD_CLEAR
                SERIAL_PRINT_NL STR_TTL1
                SERIAL_PRINT_NL STR_TTL2
                SERIAL_NL
                LCD_PRINT2      STR_TTL1, STR_TTL2
                ACALL   FIX_DELAY
                RET

INPUT:
                SERIAL_PRINT_NL STR_CNS_BRD
                SERIAL_PRINT    STR_AGE
                LCD_CLEAR
                LCD_PRINT2      STR_INP, STR_AGE
                ACALL   KP_READ
                SUBB    A, #30
                MOV     B, #10
                MUL     AB
                MOV     B, A
                ACALL   KP_READ
                SUBB    A, #30
                ADD     A, B
                MOV     INPUTAGE, A
                SERIAL_NL
                SERIAL_PRINT_NL STR_CNS_OK
                SERIAL_PRINT_NL STR_CNS_RST

                MOV     KEYPAD, #11101111B
                WAIT_KEY:
                JNB     KP_3, INPUT             ;if Reset Button is Pressed ask Input Again!
                JB      KP_1, WAIT_KEY          ;Wait Untul OK Button is Pressed
                RET
```



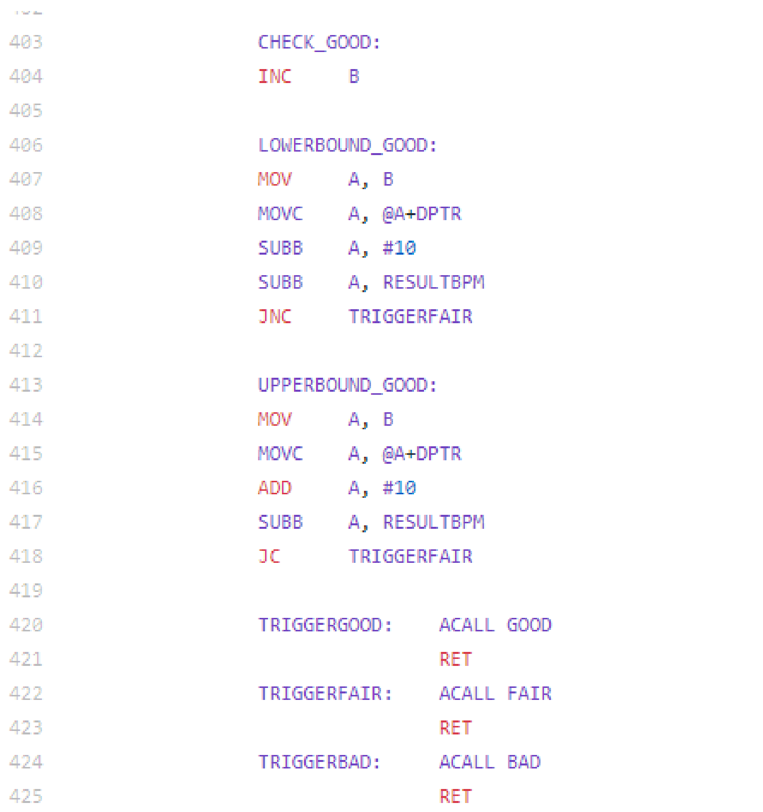
```
KEEPMSG:        SERIAL_PRINT_NL STR_CNS_LST
                LCD_CLEAR
                LCD_PRINT2      STR_LST1, STR_LST2
                RET
```



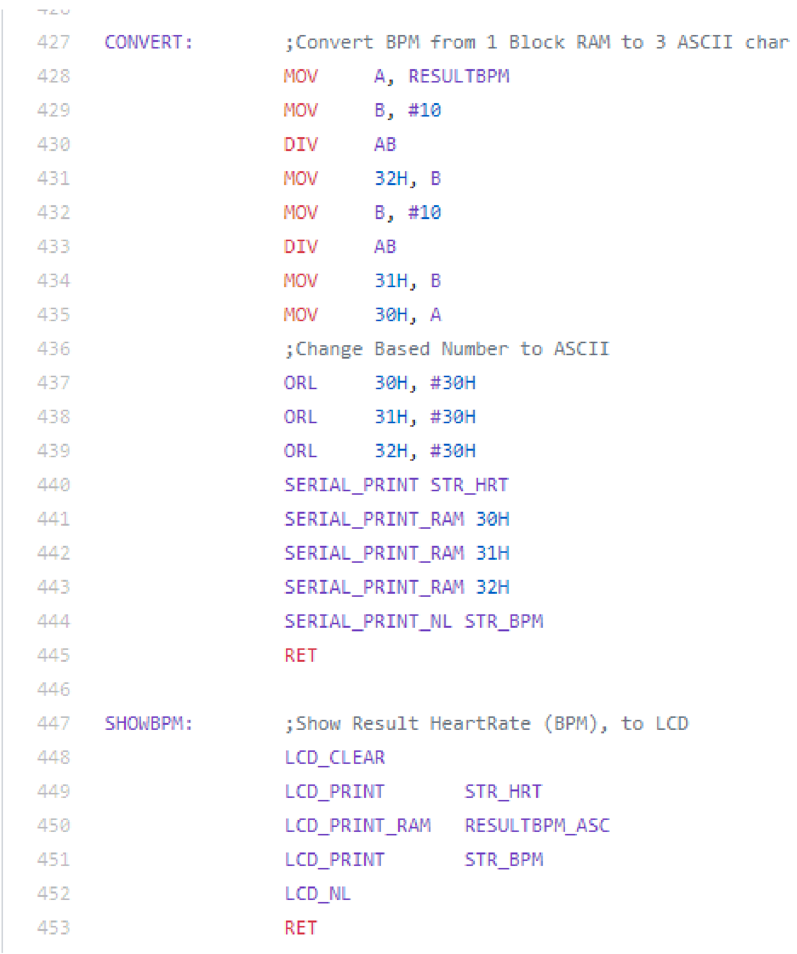
```

LISTEN:         ;Process Heartbeat Sensor
                LCD_CLEAR
                LCD_PRINT2      STR_DTT1, STR_DTT2
                SERIAL_PRINT_NL STR_CNS_CNF
                MOV     R3, #0          ;R3 is used for Counting Time
                LOWPULSE:       JNB     TF0, LOW_DETECT                 ;Low Signal Detection
                                CLR     TF0
                                INC     R3
                                CJNE    R3, #152, LOW_DETECT            ;Continue Detection if < 10 SEC
                                SJMP    DONE_10SEC
                LOW_DETECT:     JNB     HRT_SNS, LOWPULSE               ;Wait until Low Pulse
                HIGHPULSE:      JNB     TF0, HIGH_DETECT                ;High Signal Detection
                                CLR     TF0
                                INC     R3
                                CJNE    R3, #152, HIGH_DETECT           ;Continue Detection if < 10 SEC
                                SJMP    DONE_10SEC
                HIGH_DETECT:    JB      HRT_SNS, HIGHPULSE              ;Wait untul High Pulse
                CJNE    A, #0, NOWCOUNTING                              ;First Heartbeat Pulse
                ACALL   KEEPMSG                                         ;Show MSG to LCD
                MOV     A, #0                                           ;A is used for counting BPM
                MOV     TH0, #0                                         ;Set Timer Initiation from 0
                MOV     TL0, #0
                SETB    TR0                                             ;Turn on Timer on first Pulse
                NOWCOUNTING:    INC     A                               ;Increment Beat every Wave
                                SJMP    LOWPULSE                        ;Detect Again!
                DONE_10SEC:     CLR     TR0                             ;Stop Timer
                                MOV     B, #6                           ;Multiply By 6
                                MUL     AB                              ;Because we Only Scan 10 SEC
                                MOV     RESULTBPM, A                    ;Save BPM
                                RET
```



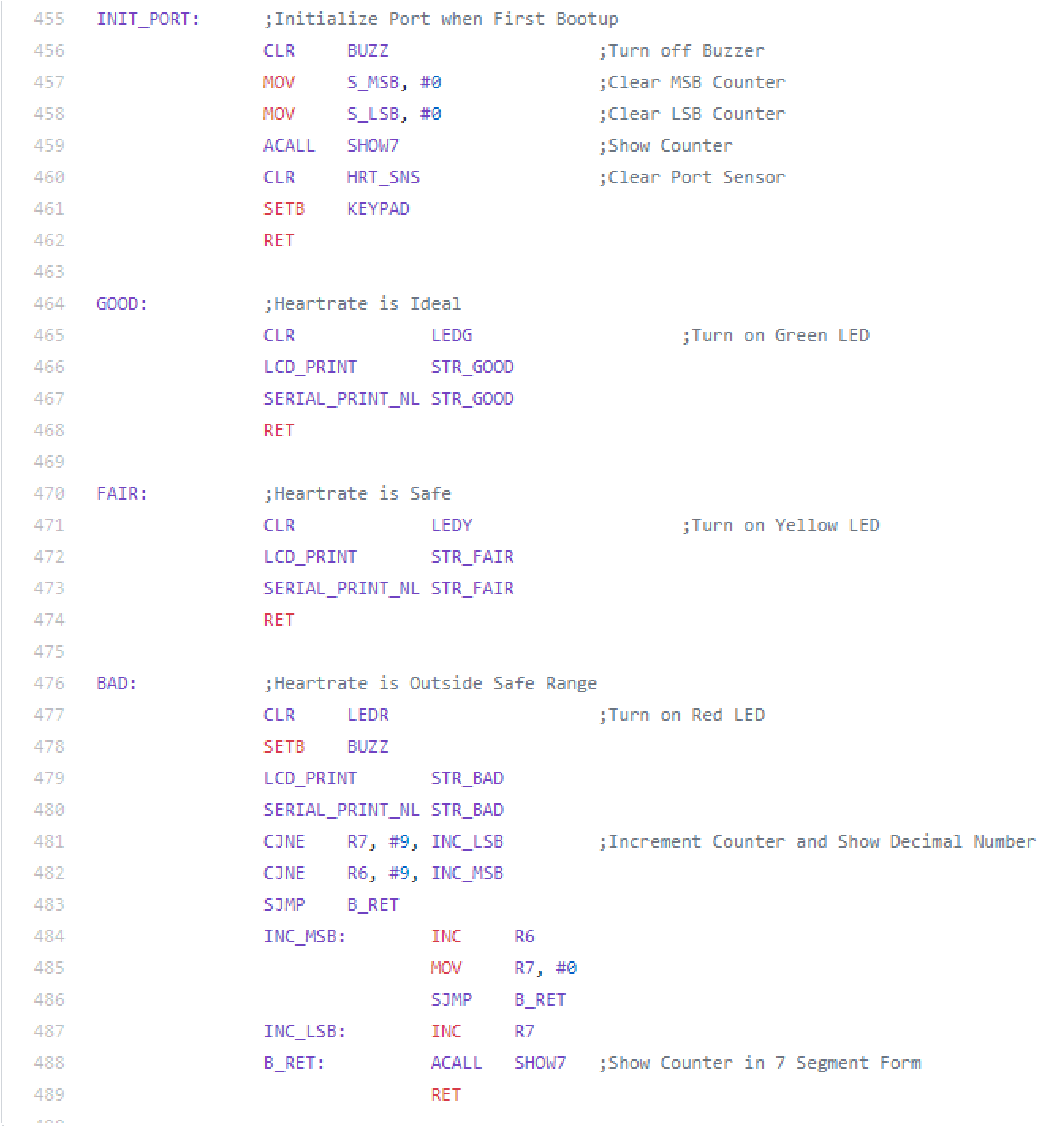
```

PROCESS_RESULT:
                SERIAL_PRINT    STR_SUM
                LCD_PRINT       STR_SUM
                MOV     A, INPUTAGE
                CJNE    A, #18, CHECKAGE
                SJMP    ABOVE18Y
                CHECKAGE:       JC      PROC
                ABOVE18Y:       MOV     A, #18
                PROC:           MOV     B, #3
                                MUL     AB
                                MOV     B, A
                                MOV     DPTR, #YEAR_RATE

                LOWERBOUND_CHECK:
                MOVC    A, @A+DPTR
                SUBB    A, RESULTBPM
                JNC     TRIGGERBAD

                UPPERBOUND_CHECK:
                MOV     A, B
                INC     A
                INC     A
                MOVC    A, @A+DPTR
                SUBB    A, RESULTBPM
                JC      TRIGGERBAD

```



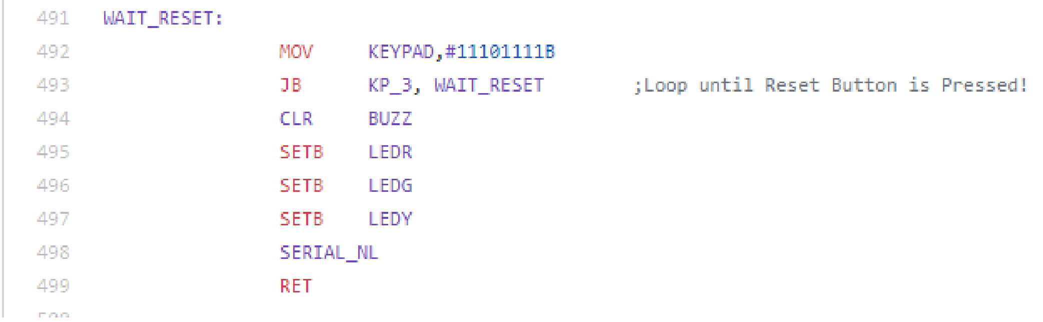
```
                CHECK_GOOD:
                INC     B

                LOWERBOUND_GOOD:
                MOV     A, B
                MOVC    A, @A+DPTR
                SUBB    A, #10
                SUBB    A, RESULTBPM
                JNC     TRIGGERFAIR

                UPPERBOUND_GOOD:
                MOV     A, B
                MOVC    A, @A+DPTR
                ADD     A, #10
                SUBB    A, RESULTBPM
                JC      TRIGGERFAIR

                TRIGGERGOOD:    ACALL GOOD
                                RET
                TRIGGERFAIR:    ACALL FAIR
                                RET
                TRIGGERBAD:     ACALL BAD
                                RET
```



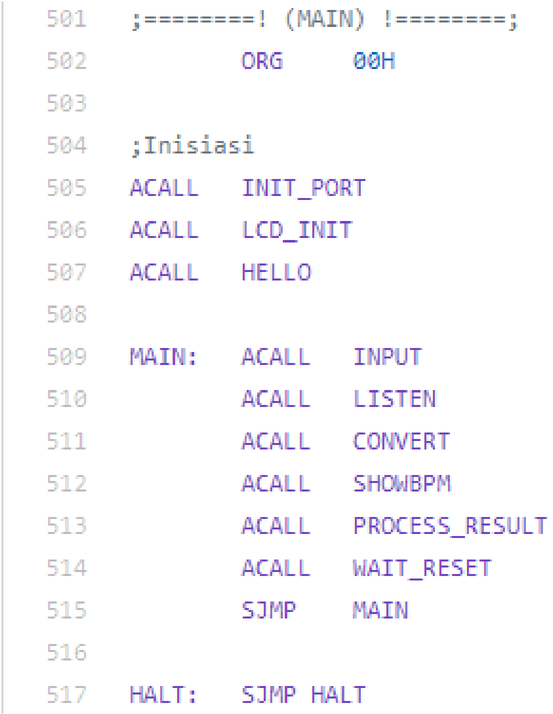
```
CONVERT:        ;Convert BPM from 1 Block RAM to 3 ASCII char
                MOV     A, RESULTBPM
                MOV     B, #10
                DIV     AB
                MOV     32H, B
                MOV     B, #10
                DIV     AB
                MOV     31H, B
                MOV     30H, A
                ;Change Based Number to ASCII
                ORL     30H, #30H
                ORL     31H, #30H
                ORL     32H, #30H
                SERIAL_PRINT STR_HRT
                SERIAL_PRINT_RAM 30H
                SERIAL_PRINT_RAM 31H
                SERIAL_PRINT_RAM 32H
                SERIAL_PRINT_NL STR_BPM
                RET

SHOWBPM:        ;Show Result HeartRate (BPM), to LCD
                LCD_CLEAR
                LCD_PRINT       STR_HRT
                LCD_PRINT_RAM   RESULTBPM_ASC
                LCD_PRINT       STR_BPM
                LCD_NL
                RET
```



```
INIT_PORT:      ;Initialize Port when First Bootup
                CLR     BUZZ                    ;Turn off Buzzer
                MOV     S_MSB, #0               ;Clear MSB Counter
                MOV     S_LSB, #0               ;Clear LSB Counter
                ACALL   SHOW7                   ;Show Counter
                CLR     HRT_SNS                 ;Clear Port Sensor
                SETB    KEYPAD
                RET

GOOD:           ;Heartrate is Ideal
                CLR             LEDG                    ;Turn on Green LED
                LCD_PRINT       STR_GOOD
                SERIAL_PRINT_NL STR_GOOD
                RET

FAIR:           ;Heartrate is Safe
                CLR             LEDY                    ;Turn on Yellow LED
                LCD_PRINT       STR_FAIR
                SERIAL_PRINT_NL STR_FAIR
                RET

BAD:            ;Heartrate is Outside Safe Range
                CLR     LEDR                    ;Turn on Red LED
                SETB    BUZZ
                LCD_PRINT       STR_BAD
                SERIAL_PRINT_NL STR_BAD
                CJNE    R7, #9, INC_LSB         ;Increment Counter and Show Decimal Number
                CJNE    R6, #9, INC_MSB
                SJMP    B_RET
                INC_MSB:        INC     R6
                                MOV     R7, #0
                                SJMP    B_RET
                INC_LSB:        INC     R7
                B_RET:          ACALL   SHOW7   ;Show Counter in 7 Segment Form
                                RET
```



```
WAIT_RESET:
                MOV     KEYPAD,#11101111B
                JB      KP_3, WAIT_RESET        ;Loop until Reset Button is Pressed!
                CLR     BUZZ
                SETB    LEDR
                SETB    LEDG
                SETB    LEDY
                SERIAL_NL
                RET
```



```
;========! (MAIN) !========;
        ORG     00H

;Inisiasi
ACALL   INIT_PORT
ACALL   LCD_INIT
ACALL   HELLO

MAIN:   ACALL   INPUT
        ACALL   LISTEN
        ACALL   CONVERT
        ACALL   SHOWBPM
        ACALL   PROCESS_RESULT
        ACALL   WAIT_RESET
        SJMP    MAIN

HALT:   SJMP HALT
```

IV. Link video presentasi

Presentasi + Penjelasan : https://youtu.be/0wvor58aK6s

Simulasi : https://youtu.be/rC1mJqsGn1Y

V. Referensi

Pemicu Masalah :

https://www.suara.com/health/2020/04/09/120000/studi-ungkap-1-dari-5-pasien-covid-19-alami-kerusakan-jantung

Heart Rate Berdasarkan Umur :

https://www.medicalnewstoday.com/articles/235710#normal-resting-heart-rate

https://www.freeprintablemedicalforms.com/preview/Resting_Heart_Rate_Chart

LCD 16x2 Pinout Diagram and Commands :

https://www.electronicsforu.com/resources/learn-electronics/16x2-lcd-pinout-diagram

Program Repository :

https://github.com/ramdanks/Abnormal_Heartbeat