```
/*
Monday, 19 May 2014
ATS Program  - Button Controlled.
uC: PIC16F628A
Fosc: 4MHz
Author: N. Chitiyo (nchitiyo@sirdc.ac.zw)
NOTE: use COFF file for debugging in Proteus ISIS
Software Rev: 0.2.2f (ATS)
Button Software Version: 0.1

Software Revisions:
0.0.1    : ATS. Auto only
0.0.2    : ATS. With Cooldown Timer
0.0.3    : ATS with key-override control
0.1.0    : software v0.0.3 for board version 0.1 (Split board design)
0.2.0    : software v0.1.0 for board version 0.2T(Split-board - Cable connected)
0.2.1    : software v0.2.0 for board version 0.2U (ULN driver version)
0.2.2    : ATS Software Cleaned up. for Documentation Purposes.
0.2.2f   : v0.2.2 implementing Feedback

Hardware Version: v0.2U (With ULN)
Hardware Revisions:
0.0      : ATS Board - Debug Version Prototype. uni-Board, Transistor Driven
0.1      : Split version. dual-board, joined by headers. Transistor Driven
0.2      : Split version dual board, joined by ribbon cable/UTP
0.2U     : dual, ULN-driven, Changeover Relay on board 1 with gen feedback


*NOTE: There's need to Edit the CoolDown Routine!! introduce a cooldown Poll
routine to fix this problem: to be looked into
*NOTE: there is need for a pre-changeover poll to verify the absence of ZESA for
 one more time, before switching GENENRATOR power to LOAD
*/

// ALL Important Functions defined and/or declared in "Initializers.c"

unsigned int ProgTimer, RealTimer, RunTimer, CoolDownTimer;
bit Auto_Flag, Run_Flag, CoolDown_Flag;
bit GenFeedFlag;         //feedback indirect register
unsigned int RunTime, CoolDownTime;
unsigned short RBValue, CrankTrials;

/*Input-Output Table:
     *  PIN  |  I/O  |  Assign                    |     Notes
     *-----------------------------------------|---------------
     *  RA0  |Output |  SQOUT/ N/A                |    Clock Count
     *  RA1  |Input  |  N/A                       |
     *  RA2  |Input  |  ZESA Sense                |
     *  RA3  |Output |  N/A                       |
     *  RA4  |Input  |  Gen Feedback              |Sense that Gen is ON
     *  RB0  |Input  |  OFF   Button              |Manual GenSTOP interrupt en.
     *  RB1  |Input  |  Start                     |Manual GenStart
     *  RB2  |Input  |  Auto                      |GenAuto
     *  RB3  |Input  |  ON                        |GenON
     *  RB4  |Output |  GenSTOP Control (NC!!!)   |
     *  RB5  |Output |  ChangeOver Control (NO)   |
     *  RB6  |Output |  GenStart Control (NO)     | crank.
     *  RB7  |Output |  GenOn Control (N.O)       |
     *-----------------------------------------|---------------
*/
sbit SQOUT at RA0_Bit;
sbit ZESA at RA2_Bit;
```

```
 63: sbit feedback at RA4_bit;
 64: sbit GenStop at RB4_Bit;
 65: sbit ChangeOver at RB5_Bit;
 66: sbit GenStart at RB6_Bit;
 67: sbit GenOn at RB7_Bit;
 68:
 69: // ///////////Declare Functions //////////////
 70: void crank();
 71: void Poll();
 72: ////////////////////////////////////////////
 73:
 74:
 75: void interrupt() {       //TMR0 Interrupt Handler
 76:         GIE_Bit = 0;
 77:         if (T0IF_Bit) {
 78:
 79:            T0IF_Bit = 0;
 80:            TMR0 = 0;
 81:            ProgTimer++;
 82:            if (progTimer == 1953)
 83:            {
 84:                RealTimer++;
 85:
 86:                //Check Run and CoolDown Status
 87:                if (RunTimer == RunTime) {
 88:                   RunTimer = 0;
 89:                   Run_flag = 0;
 90:                   CoolDown_Flag = 1;
 91:                }
 92:
 93:                if (CoolDownTimer == CoolDownTime) {
 94:                   CoolDownTimer = 0;
 95:                   //Run_flag = 1;   //The Run Flag Should Be Started Elsewhere
 96:                   CoolDown_Flag = 0;
 97:            }
 98:
 99:            if (Run_Flag && Feedback) RunTimer++;
100:            if (CoolDown_Flag) CoolDownTimer++;
101:            SQOUT=~SQOUT;
102:            progTimer = 0;
103:            }
104:         }
105:         if (INTF_Bit) {          //Pressing "Stop" forced Stop, Turn Off all
106:            INTF_Bit = 0;
107:            PORTB = 0;
108:            CrankTrials = 6;      //just to kick it out of the crank...
109:            RBValue = 19;          //so that it falls into "Default"
110:            Auto_Flag = 0;
111:            Run_Flag = 0;
112:         }
113:         GIE_Bit = 1;
114: }
115:
116:
117: void main() {
118:
119: //set the runtime (4 hours) and cooldown time (1 hour)
120:
121:       //RunTime = 20;         //Testing Purposes
122:       RunTime = 14388;   //4 hours - 12 seconds run time
123:       //CoolDownTime = 20;    //Testing Puroses
124:       CoolDownTime = 3599;   //1 hour (- 1 second) cooldown time
```

```
125:
126:       PORTA = 0;
127:       TRISA = 0b10110;               //following the Table Above
128:       PORTB = 0;
129:       TRISB = 0b00001111;            //Following the Table Above
130:       CMCON = 0x0F;                  //PORTA all digital
131:
132:       T0IE_Bit = 1;           //Enable TMR0 Interrupt
133:       INTE_Bit = 1;          //Enable RB0 Interrupt...
134:       INTEDG_Bit = 1;        //...on rising edge of RB0
135:       GIE_bit = 1;
136:
137:       // /////////Timer Configuration ////////////////////
138:
139:       T0CS_Bit = 0;    //Select Timer Mode. Timer Starts Now
140:       TMR0 = 0;        //reset the TMR0 Register
141:       PSA_Bit = 0;     //Assign Prescaler from WDT to Timer0 when value = 0
142:       OPTION_REG &=248;         //Clear Previous Prescaler Values
143:       OPTION_REG |=0;     //set Prescaler to TimeSet  (1:2)
144:
145:       ////////////////////////////////////////////////
146:
147:
148: while(1) {                            //Main Endless Loop
149:
150:      delay_ms(100);                      //delay for latency (To allow system
151:                                          //to tolerate key debounces)
152:
153:      //collect the last three values input from the ignition switch:
154:      if (RBValue != 19) RBValue = PORTB & 0x07; //19 is the fallback value
155:
156:
157:      if (!Run_flag && CoolDown_flag) RBValue = 19;      //it's time to cool down
158:
159:      /* NOTE: When the Gen Cools down, you have to Start
160:         it manually after coolddown.
161:      */
162:
163:      /* RBValue Mode Coding Table
164:                        |Auto |Start |Off |
165:                        |-----|------|----|--------------|
166:         * Condition    |RB2  |RB1   |RB0 | Hex | Dec    |
167:         *-------------|-----|------|----|-----|--------|
168:         *  Auto       | 1   | x    | 0  | 0x04| 4,6    |
169:         *  Start      | 0   | 1    | 0  | 0x02| 2      |
170:         *  OFF        | x   | x    | 1  | 0x01|1,3,5,7 |
171:         *  Maintain   | 0   | 0    | 0  | 0x00| 0      |
172:         *-----------------------------------------------
173:       */
174:
175:      switch(RBValue) {
176:
177:      // ////////////Auto Mode/////////////////
178:      case 4:                                   //Gen Auto Mode Selected
179:           Auto_Flag = 1;                     // Gen is in Auto Mode
180:           RBValue = 0;
181:           if (Auto_Flag) Poll();
182:           break;
183:
184:      case 6:                                   //Gen Auto Mode Selected
185:           Auto_Flag = 1;                     // Gen is in Auto Mode
186:           RBValue = 0;
```

```
187:           if (Auto_Flag) Poll();
188:           break;
189:
190:      // ////////////////System Start/Crank ///////////////////
191:
192:      case 2:                                    //Start Button Pressed
193:           Auto_Flag = 0;                      //Gen is in Manual Mode
194:           GenStop = 1;                         //Turn Off GEN_OFF signal
195:           GenOn = 1;
196:           crank();                             //crank till there's feedback
197:           RBValue = 0;
198:           break;
199:
200:      // ////////////////Maintain Scenario ///////////////
201:
202:      case 0:
203:           Auto_Flag = 0;
204:           RBValue = 0;
205:           break;
206:
207:      // ///////////////Off and Default Scenario//////////
208:
209:      default:
210:           Auto_Flag = 0;
211:           PORTB = 0;       //Turn the Gen Off if none of the conditions are met
212:           delay_ms(5000);
213:           if (!feedback) Run_Flag = 0; //Ensure Gen is off
214:           RBValue = 0;
215:           break;         //We need an ERROR Condition Here
216:      } //switch
217:
218:
219:  }//While
220:
221: }
222:
223: // ///Crank Part is Working!!! ////////////////////////////////////
224:
225: void crank() { //Crank till there's feedback. 5X2 seconds before failing.
226: GenFeedFlag = feedback;
227: CrankTrials = 0;
228: while (!GenFeedFlag && CrankTrials <5) {
229:    GenStart = 1;
230:    delay_ms(2000);
231:    CrankTrials++;
232:    GenFeedFlag = feedback;
233:    if (CrankTrials == 5) {
234:       CrankTrials = 0;
235:       GenFeedFlag = 1;
236:       PORTB = 0;
237:       Run_Flag = 0;
238:       RBValue = 19;  //Falls to "Default
239:       //break;
240:       }
241:    else Run_Flag = 1;
242: GenStart = 0;      //Stop Cranking
243: } //while
244: }//crank
245:
246: // //////Polling Function ///////////////////
247: void Poll() {
248:      if (Auto_Flag && ZESA) {
```

```
249:         ChangeOver = 0;    // make sure you revert to ZESA Supply
250:         delay_ms(2000);
251:         RBValue = 19;       //Fall to Default
252:         }
253:         else {             //If ZESA is not there,
254:           delay_ms(1000);    //Just wait...
255:            if (!Run_flag) {  //if the Gen is running, leave and Poll
256:               if (!CoolDown_Flag) {
257:                 GenStop = 1;
258:                 GenOn = 1;
259:                 crank();   //...and run the Generator Start Routine.
260:                 Delay_ms(5000);     //Stabilize
261:                 if (RBValue ==4 || RBValue == 6 && !ZESA) {
262:                    ChangeOver = 1;
263:                    Run_Flag = 1;
264:                    }//if AutoMode
265:                 }//if !CoolDown_Flag
266:               } // if !Run Flag
267:            }//else ZESA is not there
268: } //Poll
```