INTRODUCTION:

It is very common in houses in high terrain or high rise buildings, that the public water supply does not have enough pressure to get the water into the overhead tank. The water will be collected in a underground tank and lifted to the overhead tank by a pump. The pump must do the pumping operation whenever the water level in the overhead tank is empty and should stop pumping when filled. This operation can be performed manually by a start/stop switch for the pump motor, or the operator can control the pumping. He should constantly monitor the water level in the overhead tank to prevent overflowing. Similarly, he should ensure that there is water in the sump before starting the pump. It is very convenient to incorporate an automatic setup for pumping.

Here is a circuit that automatically controls the pump so that an efficient water supply is maintained. All possible protection features and alarms are added to this PICmicro™ microcontroller based system. The following are some of the essential features for a pump controller.

1. Pump should start pumping immediately after the water level in overhead tank is lower than the lower limit.
2. Pump should stop pumping when tank is filled.
3. Pump should not start if there is no water in the sump to prevent dry run.
4. Motor must stop if the pump is not delivering water due to any problem.
5. Motor should not run continuously more than the safe run time and it should stop after the set running time is over.
6. The pump can be controlled manually at any time by using the start/stop (manual) button.

Apart from the above, the following protective features are also incorporated.
1. Prevention of malfunctioning from fault in the sensors
2. Various alarms to indicate the status of the pumping system.

APPLICATION OPERATION:

The PICmicro™ microcontroller is provided with the following inputs to sense the various conditions:
1. Presence of water in the sump
2. Low water in overhead tank
3. Water filled in tank.
4. Pump ok

The inputs 1 to 3 can be easily obtained either by using float switches as sensor or by dipping probes as sensors in the O/H tank or in the sump. Presence of water at a level can be sensed by checking conductivity between the pair of probes erected at that level. A simple transistor and an Opto-coupler are sufficient for the input conditioning. A simple arrangement is shown in the circuit.

Outputs available form the controller are used to operate the starter of pump and to activate a buzzer.

Since the number of external lines are less than needed, the SUMP_OK input and the buzzer output are sharing the same pin. This pin, GP5, is assigned as output only when an error condition is detected.

The flow chart is rather lengthy and is not given here. The program constantly monitors all the input conditions and updates a status word in memory. If any input condition is changed, the program checks and detects if there is any malfunction. If any defects are detected, the controller sounds an alarm. If the MCU detects the need for pumping, the pump is started and checks again for the water delivered to the discharge pipe. A sensor attached to the outlet does the job. This sensor o/p goes low when the water is pumping. If the

Author: T.K. Mani
Kerala, India
email: epsilon@giasmd01.vsnl.net.in
pump is not pumping water for a period set in the program, the pump is stopped and an alarm is given. If the pumping is ok, the program checks for the other inputs to find out if the low water indication goes off after some time of pumping and again checks for time-out for pump or water full input to trip the motor.

Warning and annunciation are also provided as an audible alert to the user. Different types of interruptions are given to the audio beep produced by the buzzer, so that the status of this controller can be monitored. The beep interrupt pattern (1s and Zeros - 1 causes beep and zero no beep) can be transferred to the code register before calling the beep routine. Each beep is for a duration of 0.6 seconds and can also be varied by changing the tmr2 value.

Note: Only one error code is written in the program. This error code gives a beep pattern, (beep beep beep beep).

CIRCUIT DIAGRAM OF PUMP CONTROLLER
APPENDIX A: SOURCE CODE

;-----------------------------------------------------
;This program runs on PIC12C5XX
;This program is written as control program
;for domestic pump controller
;WRITTEN BY MANI.T.K(VU2ITI)
;-----------------------------------------------------

LIST

IFDEF __12C508
     MESSG "Processor-header file mismatch. Verify selected processor."
ENDIF

;=========================================================================
;
; Register Definitions
;
;=========================================================================

W   EQU   H'0000'
F   EQU   H'0001'

;----- Register Files -----------------------------------------------------

INDF  EQU   H'0000'
TMR0  EQU   H'0001'
PCL   EQU   H'0002'
STATUS EQU   H'0003'
PSR   EQU   H'0004'
OSCCAL EQU   H'0005'
GPIO  EQU   H'0006'
register equ   0x07
TMR1  equ   register+0
TMR2  equ   register+1
CODE  equ   register+2
COUNTER EQU   register+3
DELAY  EQU   register+4

TMR0  EQU   01
E1    EQU   0XAA

;----- STATUS Bits --------------------------------------------------------

GPWUF  EQU   H'0007'
PA0    EQU   H'0005'
NOT_TO EQU   H'0004'
NOT_PD EQU   H'0003'
Z      EQU   H'0002'
DC     EQU   H'0001'
C      EQU   H'0000'
WATER_FUL EQU   H'0000'
WATER_LO EQU   H'0001'
SUMP   EQU   H'0002'
PUMP_OK EQU   H'0003'
MANUAL EQU   H'0005'
ALARM  EQU   H'0005'
OUTPUT EQU   H'0004'

;----- OPTION Bits --------------------------------------------------------

NOT_GPWU EQU   H'0007'
NOT_GPPU EQU   H'0006'
T0CS   EQU   H'0005'
TOSE   EQU   H'0004'
PSA    EQU   H'0003'
PS2    EQU   H'0002'
PS1    EQU   H'0001'
PS0    EQU   H'0000'

© 1998 Microchip Technology Inc.
; RAM Definition

; Configuration Bits

__MAXRAM H'1F'

; IntRC_OSC EQU H'0FFE'

LIST

org 0 ; start address 0

;**************Program STARTS FROM here***************

START

BCF STATUS, GPWUF

MOVLW 0x07 ; Initialize the prescaler
MOVWF OPTION

MOVLW 0x02F ; initialize I/O lines. Only gp4 as output

TRIS GPIO ; configure the input and output

MOVF GPIO, W ;

ANDLW 0x1F ; Check for faulty inputs

XORLW 0x02 ;

BTFSC STATUS, Z 

GOTO ERROR1 ; input conditions are not ok, error bell

BTFSS GPIO, WATER_LO ; CHECK FOR LOW WATER LEVEL

GOTO PUMP_START

BTFSS GPIO, MANUAL ; Check if manual key pressed

GOTO MANUAL_MODE

BTFSS GPIO, SUMP ; Check Water in sump is low

GOTO ERROR2 ; Error. No water in sump

SLEEP

PUMP_START

BTFSS GPIO, SUMP ; verify if water in sump

GOTO START_PUMP

GOTO ERROR2 ; Error, No water in sump

START_PUMP

BSF GPIO, OUTPUT ; start pump

MOVLW 0x01 ; initialize timer0

MOVWF TMRO

MOVLW 0xFF ; Pump START test time (APP 16 SEC)

MOVWF TMR1 ; TO the TIMER1

MOVLW 0xFF ; Pump running time (APP 1 HOUR)

MOVWF TMR2 ; TO the TIMER2

LOOP

MOVF TMRO, F

BTFSC GPIO, SUMP ; check for water in sump

GOTO ERROR2

BTFSS GPIO, MANUAL ; check if manual button is pressed

GOTO MANUAL_MODE

BTFSC STATUS, Z ; Check for 0.065 sec timer

GOTO LOOP

DECFSZ TMRO, F ; Check for timeout 0f 16.56 sec

GOTO LOOP

BTFSC GPIO, PUMP_OK

DECFSZ TMRO, F

GOTO CHECK

STOP_PUMP
BCF GPIO,OUTPUT ;Switch off pump
GOTO ERROR3 ;Pump timeout error

CHECK
BTFSC GPIO,WATER_FUL ;CHECK FOR WATER FILLED
GOTO LOOP
BCF GPIO,OUTPUT
GOTO START

; ***********manual mode************
MANUAL_MODE
BTFSS GPIO,OUTPUT ;Check if pump is running
GOTO START_PUMP ; If not start pump
GOTO STOP_PUMP ; or stop pump

;*********** error routines*************
ERROR1
MOVLW E1 ;take Error code
MOVWF CODE ; now code contains the beep interrupt pattern
CALL BEEP
SLEEP
ERROR2
;
;
SLEEP
ERROR3
;
;
SLEEP
; error2, error3 etc can be similarly written
;;;*********************************************
;

; subroutine Beep. This routine outputs a beep pattern
; corresponds to the 1s in the code register
BEEP
OPTION
MOVLW 0X00F ; initialize I/O lines. GP4 AND GP5 as output
TRIS GPIO
MOVLW 0X0A
MOVWF TMR1
MOVLW 0X07
MOVWF COUNTER
REPEAT1
RLF CODE,F
DECF COUNTER
BTFSS STATUS,C
GOTO REPEAT1 ;Repeat until start of pattern (starts with first 1)
REPEAT2
BSF GPIO,ALARM
MOVF TMR0,F
BTFSS STATUS,Z
GOTO CHECK
DECFSZ TMR1,F
GOTO REPEAT2
DECFSZ COUNTER,1 ; if finished, return
RETLW 0X00
MOVLW 0X0A
MOVWF TMR1
RLF CODE,F
BTFSS STATUS,C
GOTO RSET
GOTO REPEAT2
RSET
BCF GPIO,ALARM
GOTO REPEAT2
END