Alarm Controller AN047

A Programmable Alarm System – PLS168

A basic alarm controller can be considered as a black box with several inputs and several outputs (Figure 1). Some inputs are used for detection and others for control. Detect inputs are driven from a variety of alarm transducers such as reed switches, smoke detectors, pressure mats, etc. An *ARM* input switches the system into a state which allows detection of the various alarm conditions and a *RESET* input is used to reset the system after an alarm has been triggered and dealt with or on re-entering the protected area. Outputs from the system include a sounder, a beacon and status indicators.

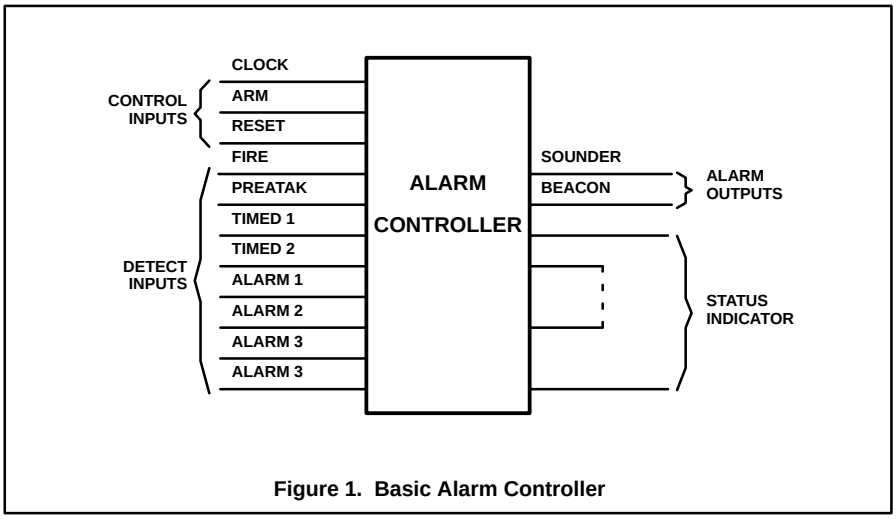

Figure 1. Basic Alarm Controller

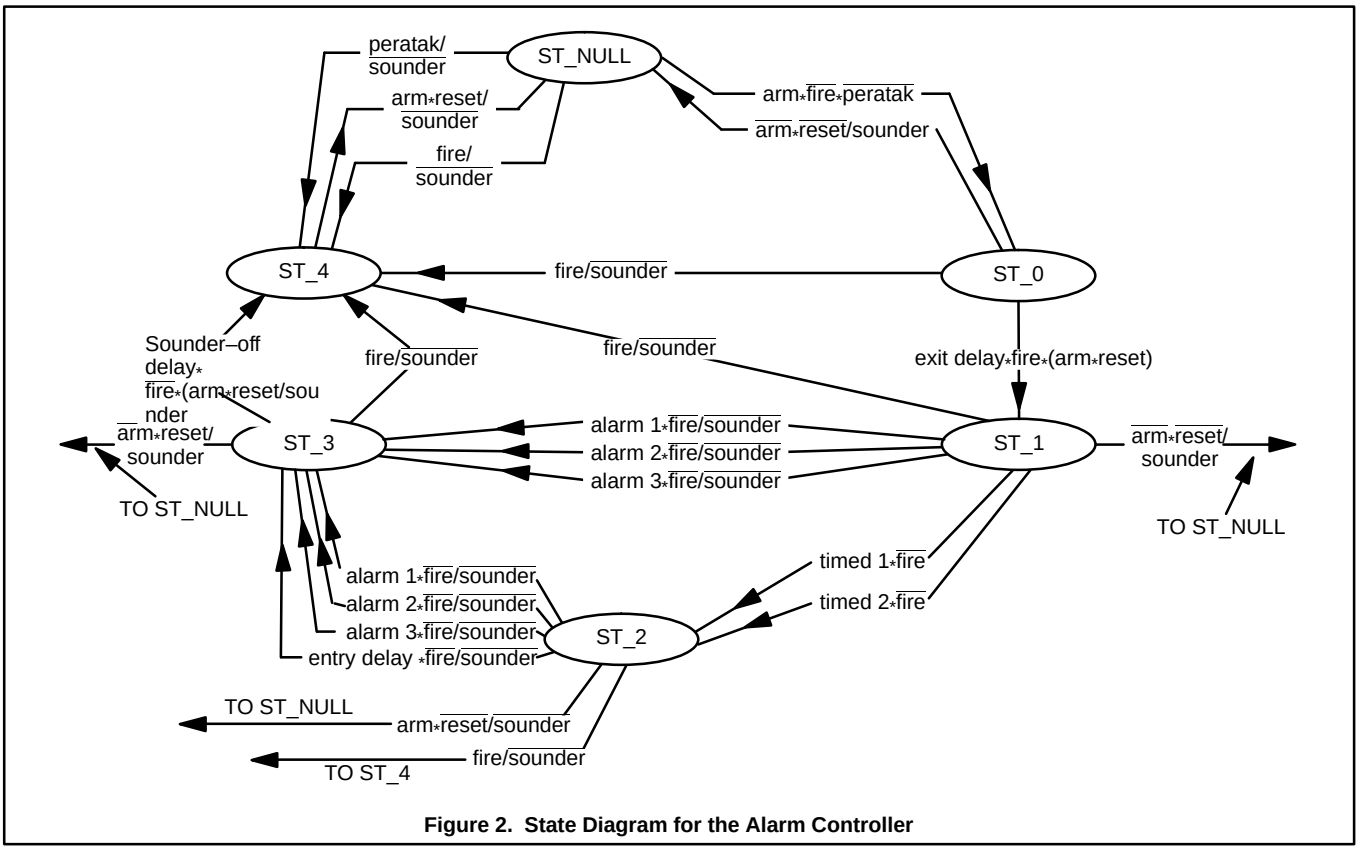

Figure 2. State Diagram for the Alarm Controller

Detect inputs can be divided into timed, untimed, fire and personal attack inputs. Timed circuits allow entry/exit delay circuits for front and rear doors, to delay operation of the alarm for approximately 16 seconds. Untimed circuits cause the alarm to operate immediately when an alarm condition occurs. These would be used to protect unusual means of entry, such as windows. Both the timed and untimed circuits should operate only if the system is armed.

The personal attack circuit is a special case untimed circuit and should operate only when the system is disarmed. The fire-detect circuit is again a special case untimed circuit and should operate regardless of whether the system is armed or not.

Outputs from the controller drive an external sounder and beacon. After 128 seconds, the sounder should turn off if the alarm has been triggered by either a timed or general untimed circuit. However, when a fire or personal attack triggers the system, the sounder should not turn off until the system is reset and the alarm condition removed.

State Machine Implementation

This design is best implemented as a state machine. The state diagram is derived from the verbal system description. Please note from Figure 2 the controller can be in one of six possible states. Examine the transitions from *ST_NULL* as an example. If a personal attack or fire condition occurs while in this state, a transition to *ST_1* takes place as indicated by the arrows on the diagram. Also at this time the sounder and beacon are activated, thus giving the alarm. If the fire and personal attack conditions have not occurred and the *ARM* SWITCH is set, then a transition to *ST_0* takes place.

Similarly, other arrows on the state diagram represent transitions between other states when specified input conditions occur. Output parameters are shown to the right of the slash line. Where there are no output parameters specified in a transition term, this indicates that no output changes are desired during this transition. That is, an output will hold its present value until told to change.

PLD Implementation

Having defined the desired system operation it is now time to select the required device to implement the desired system function from the PLD Data Manual. In this case, the device selected is the PLS168. Figure 3 shows the pinning information for the alarm controller. A 10-bit counter within the controller produces the entry/exit and sounder turn-off delays since this makes more efficient use of the PLD facilities than implementing the delays as part of the state machine. This counter uses seven internal registers with feedback and three without. For those registers without feedback, external wiring feeds their outputs back into the device to complete the 20-bit counter. Pins five to ten are used for this purpose. Output T7 also forms part of the counter.

Three other registers form the state registers and are labeled SR0, SR1 and BEACON. State vectors for these registers have to be chosen with care to ensure that the beacon output is activated at the correct time. Other inputs and outputs are as already discussed. Note that the PR/OE pin is not used. SNAP defaults its use to a register PRESET function. This pin should be tied to ground in the final circuit.

The EQN file of SNAP is separated into sections. First, in the @PINLIST section all of the signal names connected directly to the pins and their function is listed. If a signal name is used later in the file and not listed in the @PINLIST section, that signal is assumed to represent an internal node. The @PINLIST and @LOGIC EQUATIONS sections of the EQN file are shown in Table 1. The remaining state machine portion of the EQN file is shown in Table 2. Register SR0 halts and clears the counter while the controller is in certain states. This needs to be considered when defining the state vectors.

TABLE 1. SNAP EQUATIONS

```
"-----------------------------------------"
"           ALARM CONTROLLER              "
"-----------------------------------------"
@PINLIST
clock i;
arm   i;
reset i;
peratak i;
t8in  i;
t9in  i;
t10in i;
alarm3 i;
alarm2 i;
alarm1 i;
timed2 i;
timed1 i;
fire   i;

t10     o;
t9      o;
t8      o;
t7o     o;
sounder o;
beacon  o;
sr1     o;
sr0     o;

@GROUPS
@TRUTHTABLE
@LOGIC EQUATIONS
 "ten-bit counter for delay"

t1.s = /t1*/sr0;
t1.r =  t1*/sr0 + sr0;
t1.clk = clock;
t2.s =  t1*/t2*/sr0;
t2.r =  t1* t2*/sr0 + sr0;
t2.clk = clock;
t3.s =  t1* t2*/t3*/sr0;
t3.r =  t1* t2* t3*/sr0 + sr0;
t3.clk = clock;
t4.s =  t1* t2* t3*/t4*/sr0;
t4.r =  t1* t2* t3* t4*/sr0 + sr0;
t4.clk = clock;
t5.s =  t1* t2* t3* t4*/t5*/sr0;
t5.r =  t1* t2* t3* t4* t5*/sr0 + sr0;
t5.clk = clock;
t6.s =  t1* t2* t3* t4* t5*/t6*/sr0;
t6.r =  t1* t2* t3* t4* t5* t6*/sr0 + sr0;
t6.clk = clock;
t7.s =  t1* t2* t3* t4* t5* t6*/t7*/sr0;
t7.r =  t1* t2* t3* t4* t5* t6* t7*/sr0 + sr0;
t7.clk = clock;
t7o=t7;
t8.s =  t1* t2* t3* t4* t5* t6* t7*/t8in*/sr0;
t8.r =  t1* t2* t3* t4* t5* t6* t7* t8in*/sr0 + sr0;
t8.clk = clock;
t9.s =  t1* t2* t3* t4* t5* t6* t7* t8in*/t9in*/sr0;
t9.r =  t1* t2* t3* t4* t5* t6* t7* t8in* t9in*/sr0 + sr0;
t9.clk = clock;
t10.s=  t1* t2* t3* t4* t5* t6* t7* t8in* t9in*/t10in*/sr0;
t10.r=  t1* t2* t3* t4* t5* t6* t7* t8in* t9in* t10in*/sr0 + sr0;
t10.clk = clock;

sr0.clk = clock;
sr1.clk = clock;
beacon.clk = clock;
sounder.clk = clock;
```

(EQN file continued in Table 2)

State Equation Entry

The state equation entry portion of the EQN file uses a state-transition language, parameters of which are taken directly from the state diagram. Information is entered into this file in a free format. The only points to remember are that the square brackets should be used throughout to define the state registers and transitions, semicolons should be used to mark the end of vector definition. State vectors can be defined in the state equation entry file as shown in Table 2. State vectors are simply a means of labeling an arrangement of state registers which can be used later to define state transitions. Because we are using the *BEACON* output register as a state register also and SR0 is being used to halt and clear the 10-bit counter, particular care must be taken in defining the state vectors in this instance.

From the state diagram, the counter must begin counting during states *ST_0, ST_2* and *ST_3* and it must be cleared during states *ST_1, ST_4* and *ST_NULL*. State *ST_NULL* represents the power-up state of the PLS168 in which all register outputs are at logic one. Thus the inactive state of the counter is defined as being when SR0 is at logic one, therefore, SR0 must be at this level during states *ST_1* and *ST_4* and at logic zero during other states. The alarm beacon is considered to be active by an active-low signal and must be activated during states *ST_3* and *ST_4*. Register SR1 must therefore be chosen to ensure mutual exclusivity between state vectors. Input and output vectors can be defined in the same manner in terms of input and output label names. In this case, however, the label names are used directly. State transitions can now be derived directly from the state diagram. Entry/exit and sounder turn-off delay times are represented as a decoding of the 10-bit counter states. Thus to get the desired 16 second entry/exit delay. t7 must be decoded and to achieve the 128 second sounder turn-off delay t10in must be decoded.

TABLE 2. SNAP EQUATIONS

```
@INPUT VECTORS
@OUTPUT VECTORS
[sounder]srff
s_on  = 0b;
s_off = 1b;

@STATE VECTORS
[sr0, sr1, beacon]srff
st_null = 111b;
st_0    = 001b;
st_1    = 101b;
st_2    = 011b;
st_3    = 010b;
st_4    = 100b;

@TRANSITIONS

while [st_null]
 if [arm*/fire*/peratak]   then [st_0]
 if [peratak]              then [st_4]    with [s_on]
 if [fire]                 then [st_4]    with [s_on]

while [st_0]
 if [t7*/fire*(arm+reset)] then [st_1]
 if [/arm*/reset]          then [st_null] with [s_off]
 if [fire]                 then [st_4]    with [s_on]

while [st_1]
 if [timed1*/fire]         then [st_2]
 if [timed2*/fire]         then [st_2]
 if [alarm1*/fire]         then [st_3]    with [s_on]
 if [alarm2*/fire]         then [st_3]    with [s_on]
 if [alarm3*/fire]         then [st_3]    with [s_on]
 if [/arm*/reset]          then [st_null] with [s_off]
 if [fire]                 then [st_4]    with [s_on]

while [st_2]
 if [t7*/fire]             then [st_3]   with  [s_on]
 if [alarm1*/fire]         then [st_3]   with  [s_on]
 if [alarm2*/fire]         then [st_3]   with  [s_on]
 if [alarm3*/fire]         then [st_3]   with  [s_on]
 if [/arm*/reset]          then [st_null] with [s_off]
 if [fire]                 then [st_4]    with [s_on]

while [st_3]
 if [t10in*/fire*(arm+reset)] then [st_4] with [s_off]
 if [/arm*/reset]          then [st_null] with [s_off]
 if [fire]                 then [st_4]    with [s_on]

while [st_4]
 if [/arm*/reset]          then [st_null] with [s_off]
```

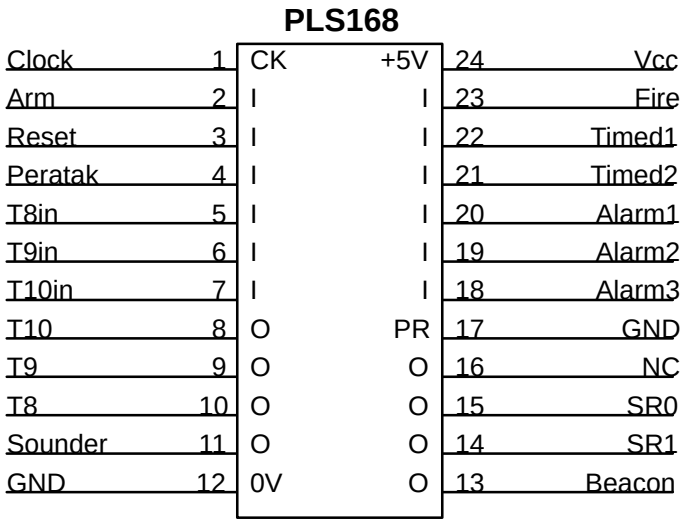

Figure 3. Pinning Information for the Alarm Controller

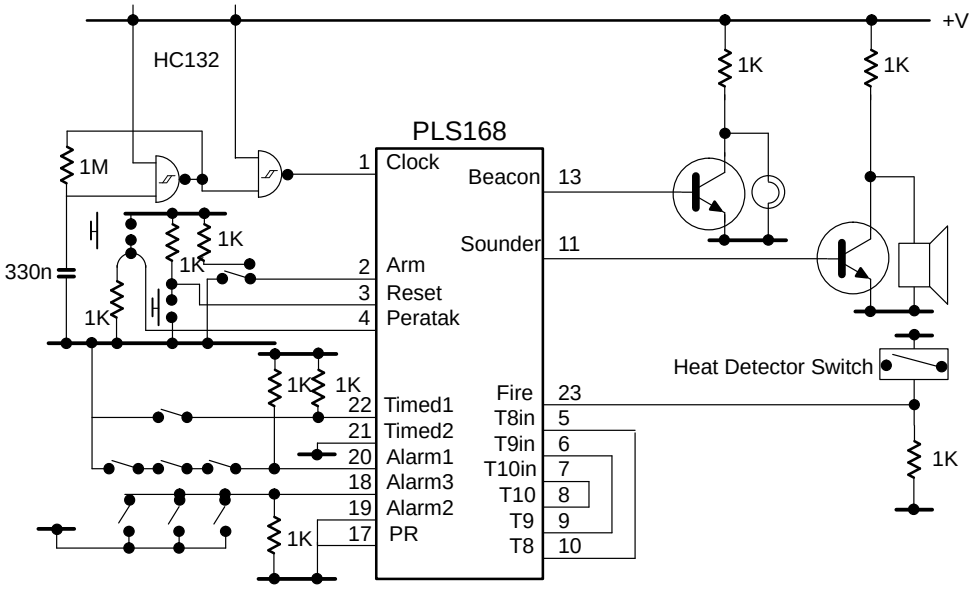

Figure 4. Alarm System based on the PLS168

With the system fully defined, simply assemble the design information using SNAP.

Functioning of the device can be verified with the SNAP simulator, which can also be used to check A.C. timings before downloading the pattern to a device programmer.

Programmability

The PLS168 device could now be used as the controller of an alarm system. As it stands, the device assumes that all the alarm inputs indicate an alarm condition when in the high state, logic one, and that the alarms are activated when the alarm outputs are active low (i.e., at logic zero).

Should an alarm input transducer be used which indicates an alarm condition as a low state, this can be catered for by altering the EQN file. For example, consider a smoke detector which outputs logic zero on detection of an alarm condition and assume that this transducer is driving the “fire” input of the device. By changing all references to ‘fire’ in the EQN file to ‘/fire’ and all instances of ‘/fire’ to ‘fire’ then the activation of the alarms will occur when logic zero is applied to this input and not when logic one is applied, as in the original case.

Polarity of the output signals cannot be altered as easily, as the device will always power-up with the outputs at logic one. This should not prove to be a problem since the outputs simply drive output transistors and

these can be used to produce the correct polarity signal for the beacon and sounder.

System Implementation

Figure 2 shows a typical alarm system based on this device. The system clock is produced by a relaxation oscillator built from 74HC132 Schmitt Triggers. Values of R_1 and C_1 shown result in a frequency of approximately 4Hz which will provide the desired entry/exit and sounder turn-off delays. These delays can be modified either by changing the external oscillator circuit or by decoding a different internal counter state. For example, to increase the entry/exit delay change all references to t7 in the EQN file to t8. Both normally-closed and normally-open loop implementations are shown. Due to the distances involved in an alarm system, the open-loop configuration may cause problems, being driven by the positive supply. to avoid this problem, input-detect polarity of the open-loop circuit can be changed by altering the EQN file.

Status indication can be provided by connecting LEDs as in Figure 5. When the reset button is pressed, any LED being lit will indicate an alarm condition for that input. This will not reset the alarm system unless the arm switch is off.

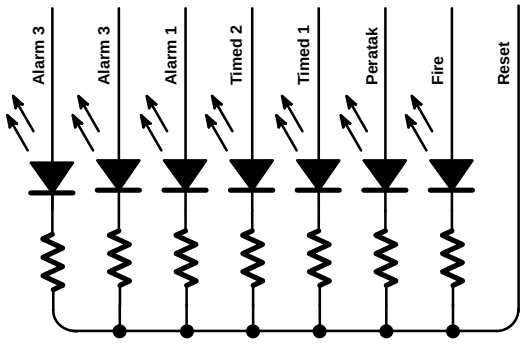

Figure 5. Status LEDs Connected to the alarm controller as shown provide status information