

Code

<u>Reset</u>	<u>Processor Initialization</u>
<u>Main</u>	<u>Main Loop to show lights</u>
<u>Main Advance</u>	<u>Advances to the next lights</u>
<u>RefreshCol</u>	<u>Regresses a single column of lights</u>
<u>Calc Times</u>	<u>Calculates time to put lights on</u>
<u>Init/Refresh</u>	<u>Basic Init and Refresh routines for table based patterns</u>
<u>Advance</u>	<u>Basic advance routine for table based patterns</u>
<u>Init Random</u>	<u>Initialization routine for the random patterns</u>
<u>Refresh Random</u>	<u>Refresh routine for the random patterns</u>
<u>Advance Random</u>	<u>Advance routine for the random patterns</u>
<u>Init/Refresh Drip</u>	<u>Init /Refresh routines for the Drip pattern</u>
<u>Advance Drip</u>	<u>Advance routine for the Drip Pattern</u>
<u>Reset Drip</u>	<u>Reset portion of the Drip Advance</u>
<u>Drip Fill</u>	<u>Fill portion of the Drip Advance</u>
<u>Drip Fill 2</u>	<u>Remainder of fill portion of the Drip Advance</u>
<u>Drip Drain</u>	<u>Drain portion of the Drip Advance</u>

Data Tables

<u>Scheme Table</u>	<u>Defines the basic tree patterns</u>
<u>Call Scheme Table</u>	<u>Used for dynamic code based patterns</u>

The core of the Tree 1.1 logic is based on a table of all the patterns (referred to as Schemes here) to display. There are basically two types of patterns:

- 1) Table based which have a sequence of steps that it iterates through which have a starting on/off state and ending on/off state for each LED.
- 2) Call based in which the logic for the state of each LED is determined by code.

Both of these are mixed in a single LOOKUP_SCHEMADATA table which provides the sequence of schemes to be displayed when you press the change pattern button. To make it easier to define, we have two macros: DEFSCHEME and DEFCALLSCHEME which take a number of parameters and create the appropriate table entries. The basic syntax is:

```
DEFSCHEME PAT_{name},ITER,TIME,TAB_{name},S1,S2,S3,S4,PAT_{next},FLAGS
DEFCALLSCHEME PAT_{name},ITER,TIME,TAB_{name},PARAM1,PARAM2,PAT_{next},FLAGS
```

Where:

- {name}** The name assigned to the pattern. This is assigned the pattern index automatically by the macro.
- ITER** How many iterations to run the full pattern for
- TIME** How long an individual refresh cycle is for. This controls the speed of the glow
- TAB_{name}** Start of the table defining the pattern
- S1,S2,S3,S4** Offsets from the start of the table where each column pattern starts
- PARAM1,PARAM2** Scheme specific values to pass to the functions
- PAT_{next}** Scheme to follow this in autocycle mode
- FLAGS** Parameters to control running this pattern, they are one or more of
 - F_NOFADE** Don't fade lights, just do solid on/off
 - F_NOWRAP** Don't wrap the column patterns individually, reset them all
 - F_DOSLEEP** Go to sleep after this pattern finishes
 - F_STARTUP** This is the startup pattern

There are also two corresponding tables that contains the actual data for the patterns which the TAB_{name} refers to. The total number of entries corresponding to the pattern are found by looking for the TAB_{name}9999 label in the same table. To take a simple case, we can look at the startup pattern which we run as part of the POST process. The scheme entry is pretty simple with

```
DEFSCHEME PAT_STARTUP,1,.100,TAB_STARTUP,0,0,0,0,PAT_MARQUIS,M_STARTUP|M_NOWRAP
```

Which tells us we are to run through all the column patterns once taking 100 cycles to do it (approximately 1.25 seconds) with all column patterns at the base of the table. When the pattern advances, it goes to the Marquis pattern. Additionally, this is the startup pattern and if we get past the end of the data for any column, we reset back to the start of the pattern for all of the columns. The column pattern table consists of a single entry in each of the tables as follows

```
STARTTABLE LOOKUP_INITIAL
TAB_STARTUP
TAB_STARTUP9999
RETLW B'11111111' ; 0 - Off, then turn on
```

and the toggle data is:

```
STARTTABLE LOOKUP_TOGGLE
TOG_STARTUP
RETLW B'00000000' ; 0 - Off, then turn on
```

What this tells us to do is at the start the lights are all off and over the next 100 cycles they progressively get brighter (by 200/100 or 2 at a time) ending up at full on at the end of the pattern.

Call schemes are a bit more sophisticated in that they can set the patterns to whatever they want. For these schemes, there are exactly two entries in the two tables. The lookup_initial table entries are the initialization and the display routines. The lookup_toggle entries are the advance routine and an unused entry. There are 4 different call schemes on the tree but three of them actually share the same routines. The independent independent scheme is Drip while there are three schemes which share the Random routine. Looking in the scheme table for the random routines, you find:

```

DEFCALLSCHEME PAT_GLOW, .209, .1, TAB_RANDOM, M_DARKCYCLE|B'111', .10, PAT_ONOFF, 0
DEFCALLSCHEME PAT_BLINKING, .209, .1, TAB_RANDOM, M_DARKCYCLE|B'111', .
    10, PAT_FLASHING, M_NOFADE
DEFCALLSCHEME PAT_FLASHING, .209, .1, TAB_RANDOM, M_DARKCYCLE|B'1100', .
    10, PAT_SWING, M_NOFADE

```

For all three patterns, we have an iteration count of 209 and a second parameter of 10. This is used by the pattern to run 10 times before it indicates it has completed an iteration. The main code iterates a total of 209 times giving us 2090 iterations before we would automatically advance to the next pattern. Since there are approximately 80 iterations per second, this means that the pattern would run for 26 seconds in automatic mode.

The first parameter which is of the form `M_DARKCYCLE|B'xxx'` is used by the pattern to determine how fast to cycle the lights. It is used as a mask against a random number to determine how much to increment/decrement the light at a time. The Glow and Blinking patterns have a mask of `B'111'` (7) to add between 1 and 8 to a light each time meaning that a light could blink as slow as once ever 6.4 seconds or as fast as every .8 seconds for a nice slow pattern. For the Flashing pattern with a mask of `B'1100'` (12), it will add either 4, 8, 12, or 16 to the light every time giving a pattern that blinks as slow as once ever 1.6 seconds or as fast as 0.4 seconds (2.5 times / second).

The `M_NOFADE` for the Flashing pattern also tells it not to glow the lights on, but to just have them in either an on or an off state.

Lastly all three of these patterns use `TAB_RANDOM` to find the called routines. This is defined as

```

    STARTTABLE    LOOKUP_INITIAL
...
TAB_RANDOM
    GOTO INIT_RANDOM
TAB_RANDOM9999
    GOTO SHOW_RANDOM

```

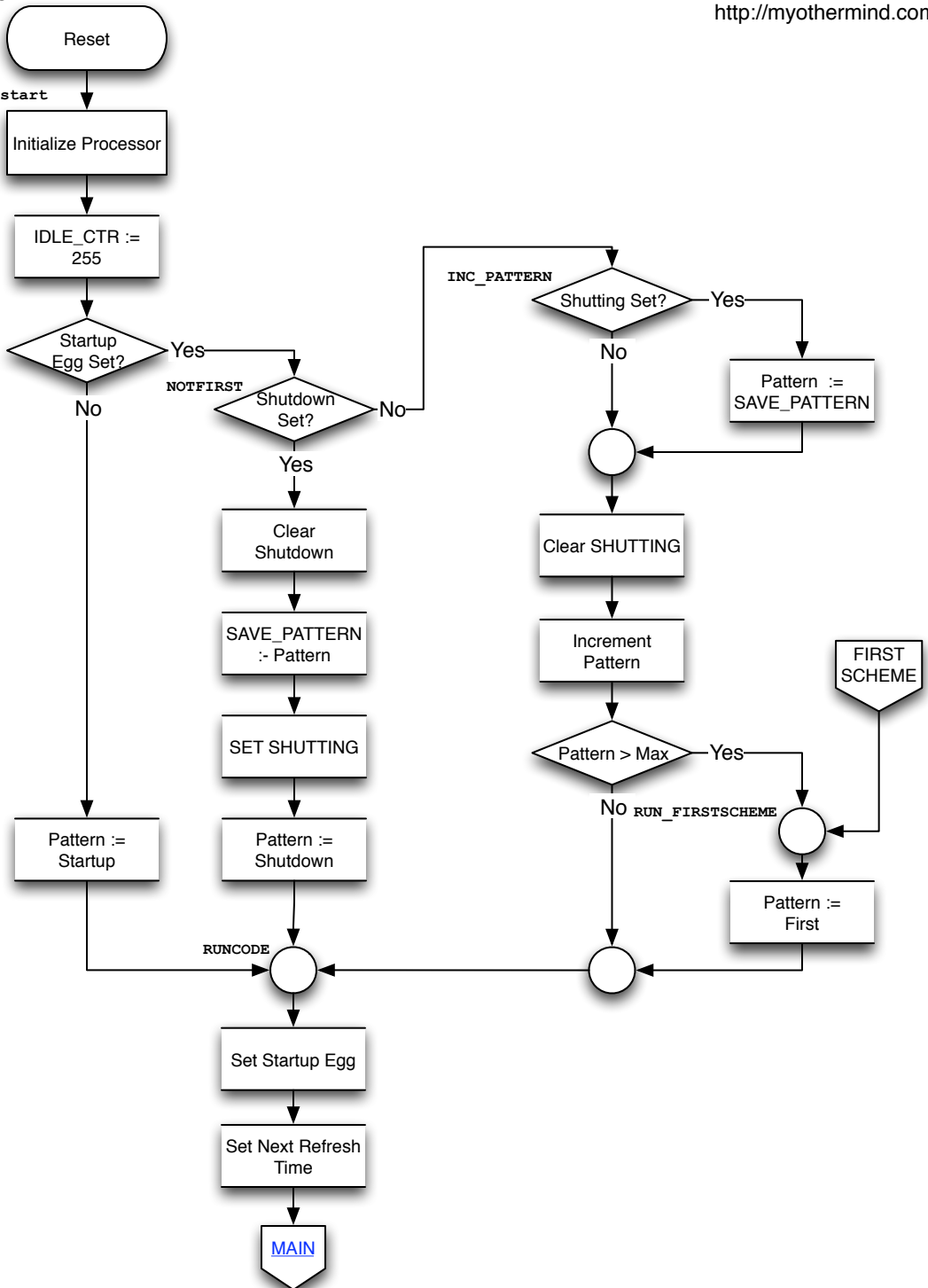
and the toggle data is:

```

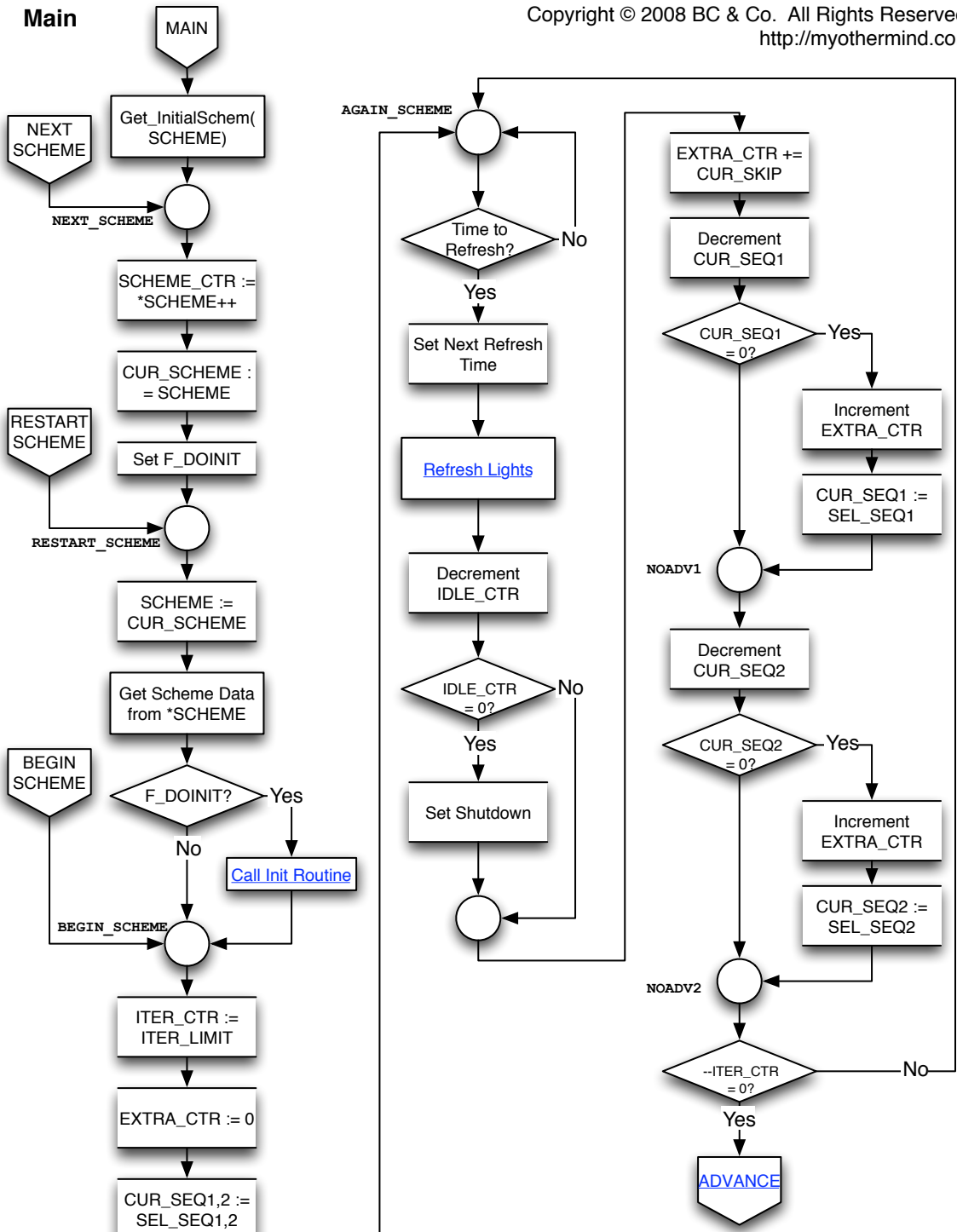
    STARTTABLE    LOOKUP_TOGGLE
...
TOG_RANDOM
    GOTO ADVANCE_RANDOM
    RETURN

```

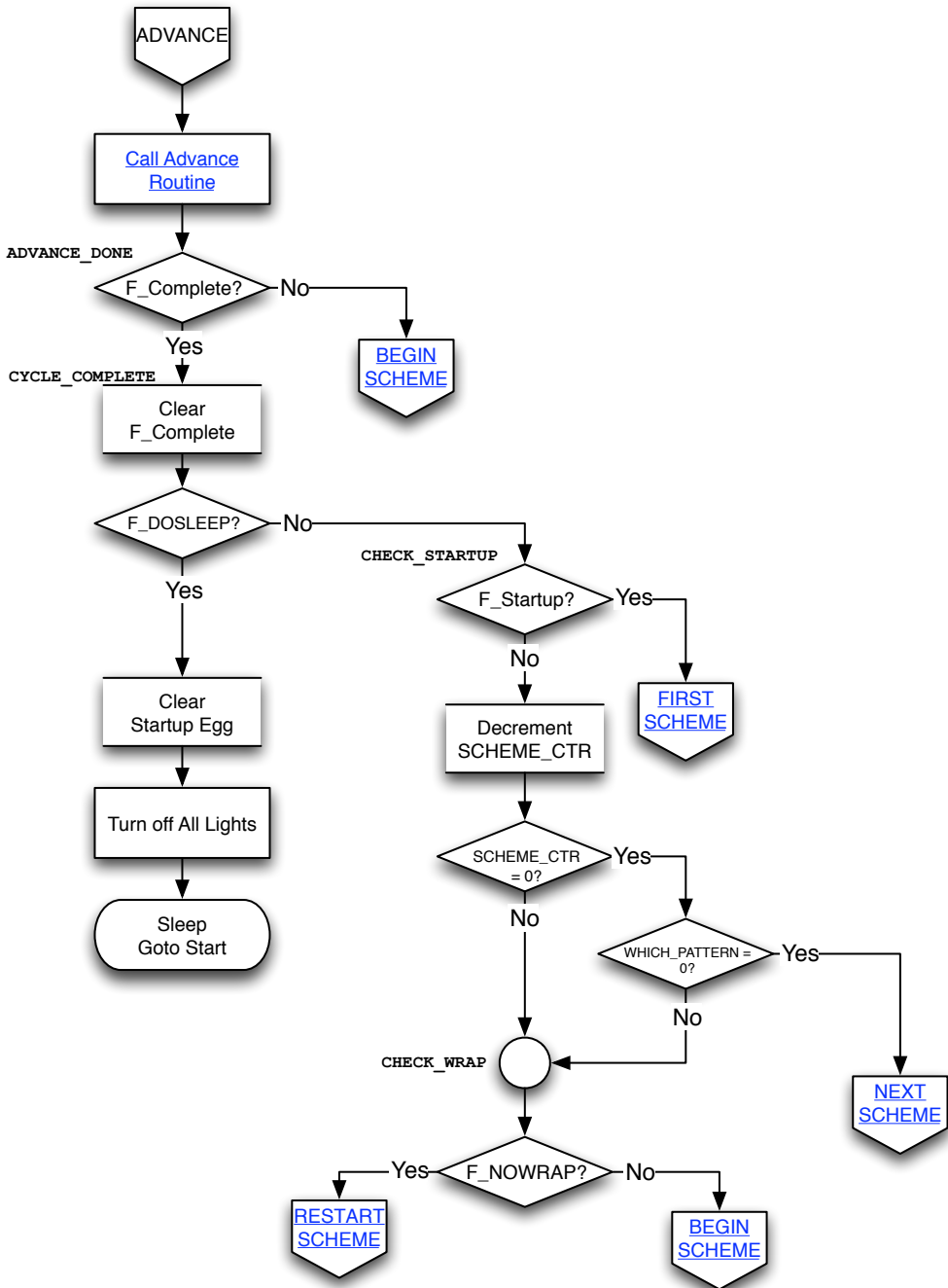
Reset

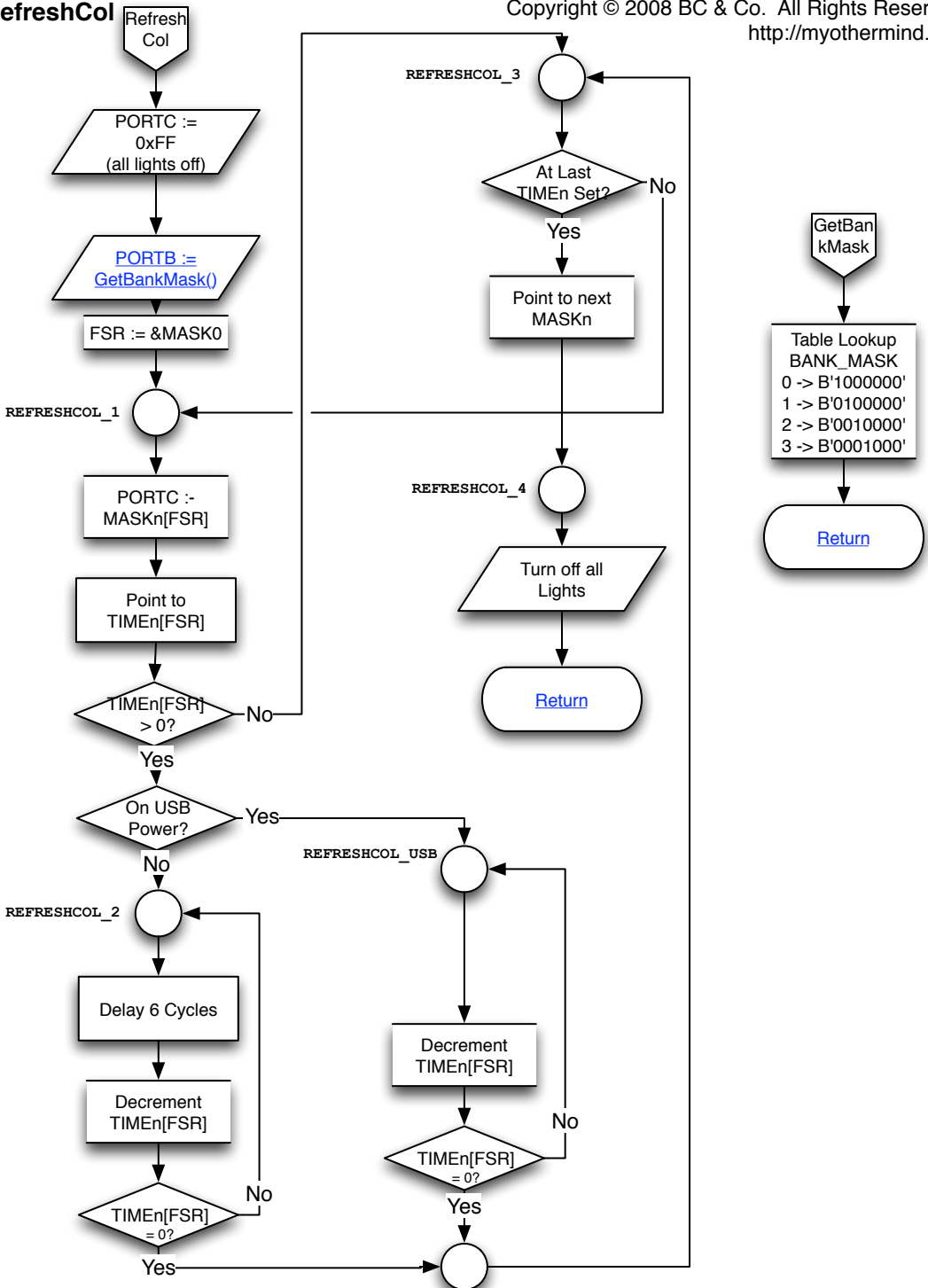


Main

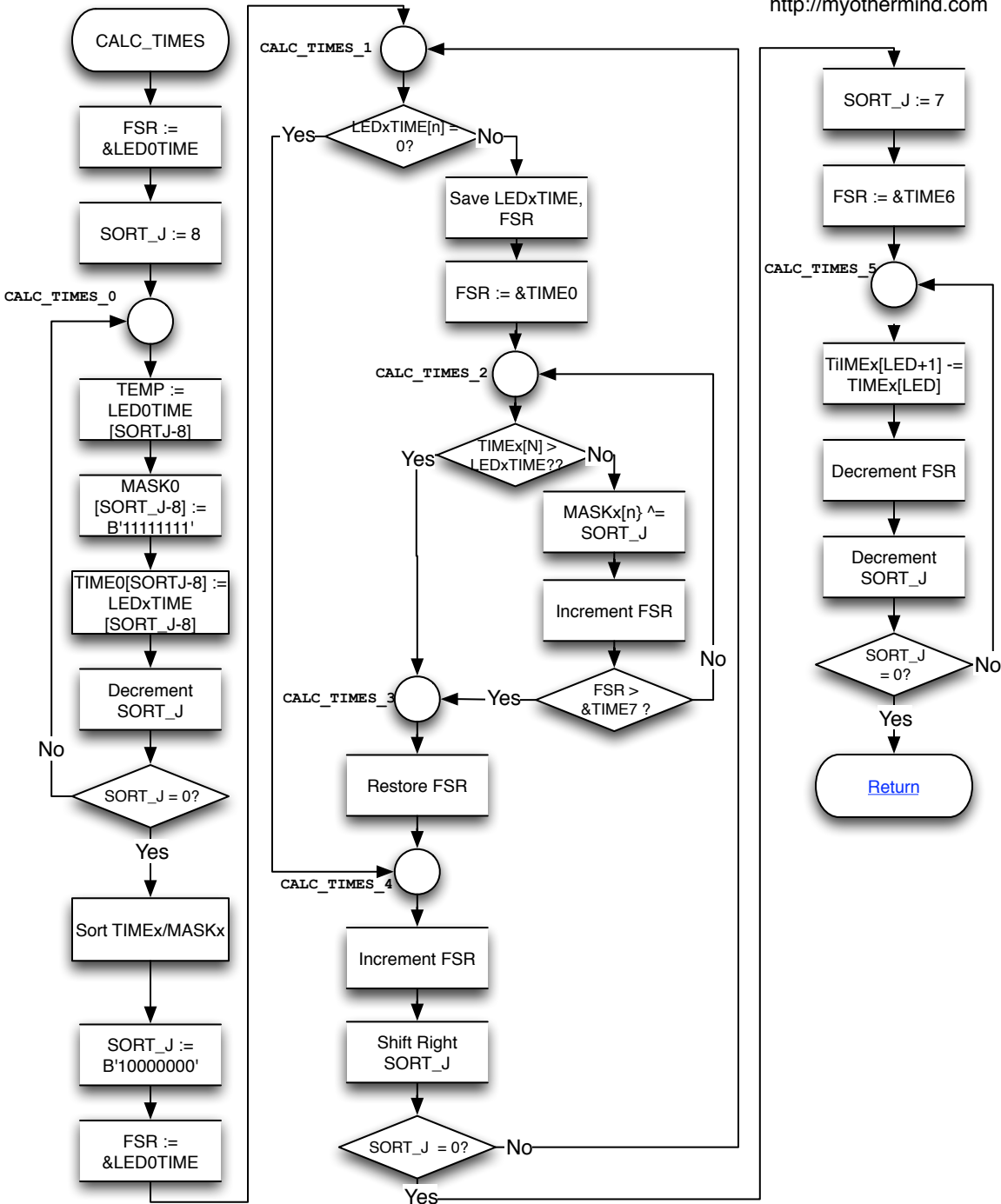


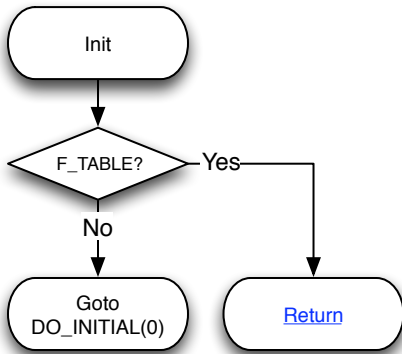
Main Advance



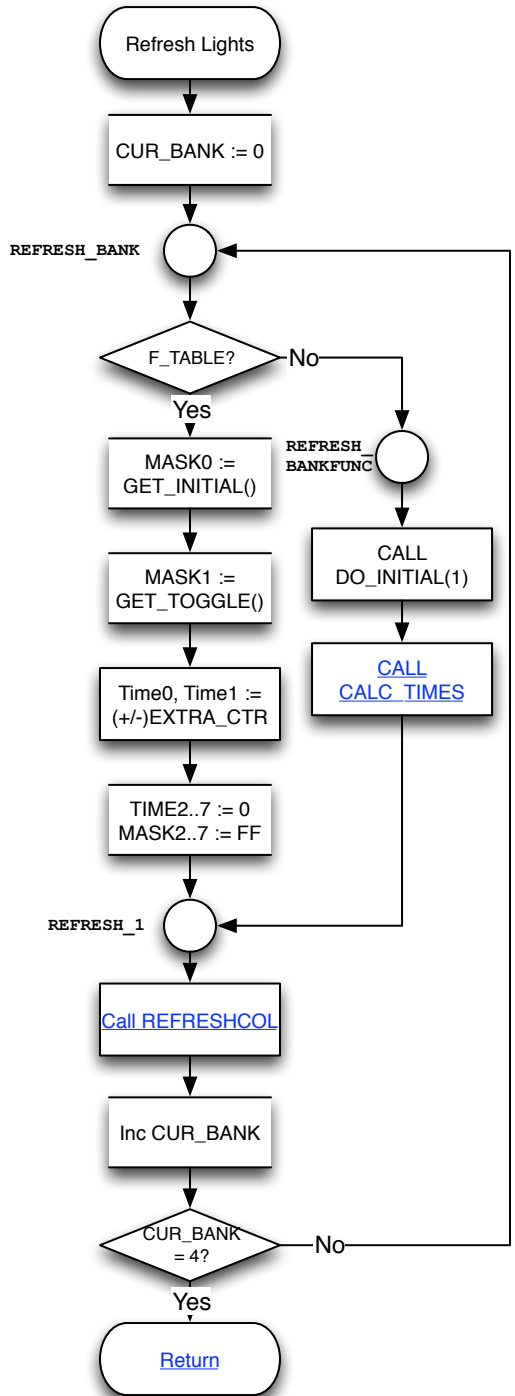


Calc Times





Note that we can optimize this by always calling DO_INITIAL(BANK0) since it has no side effects in the table case



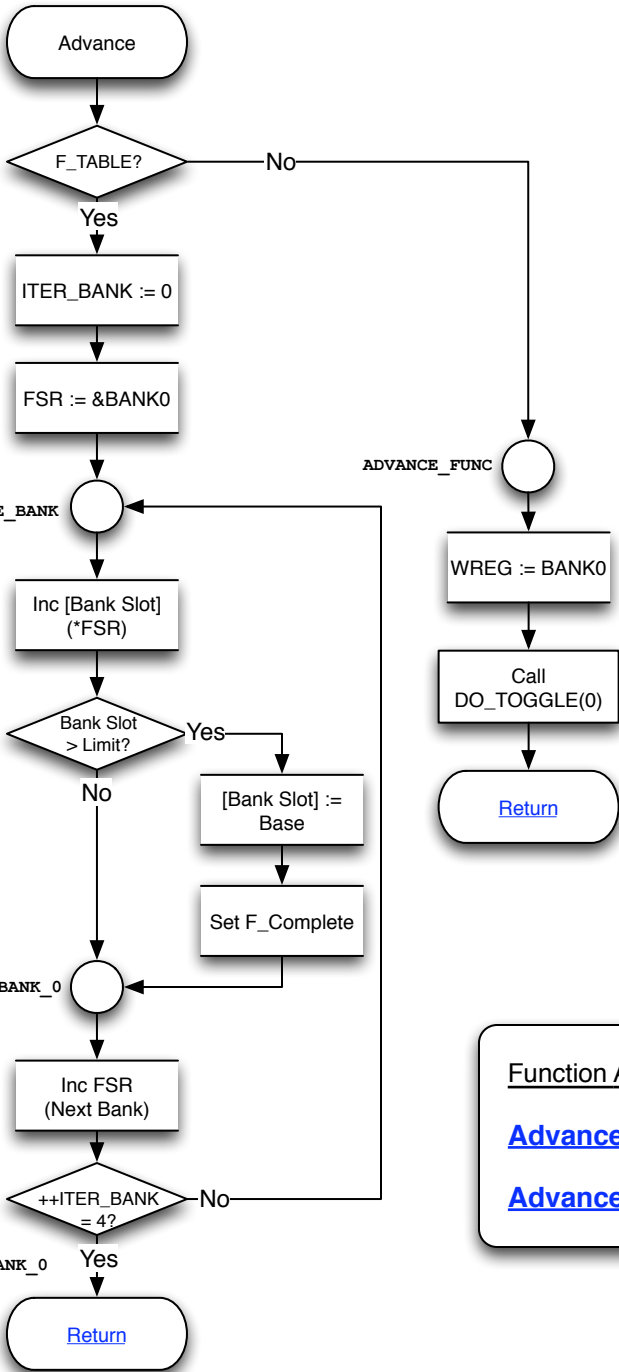
Function Init/Refresh Routines

[Init Random](#)

[Refresh Random](#)

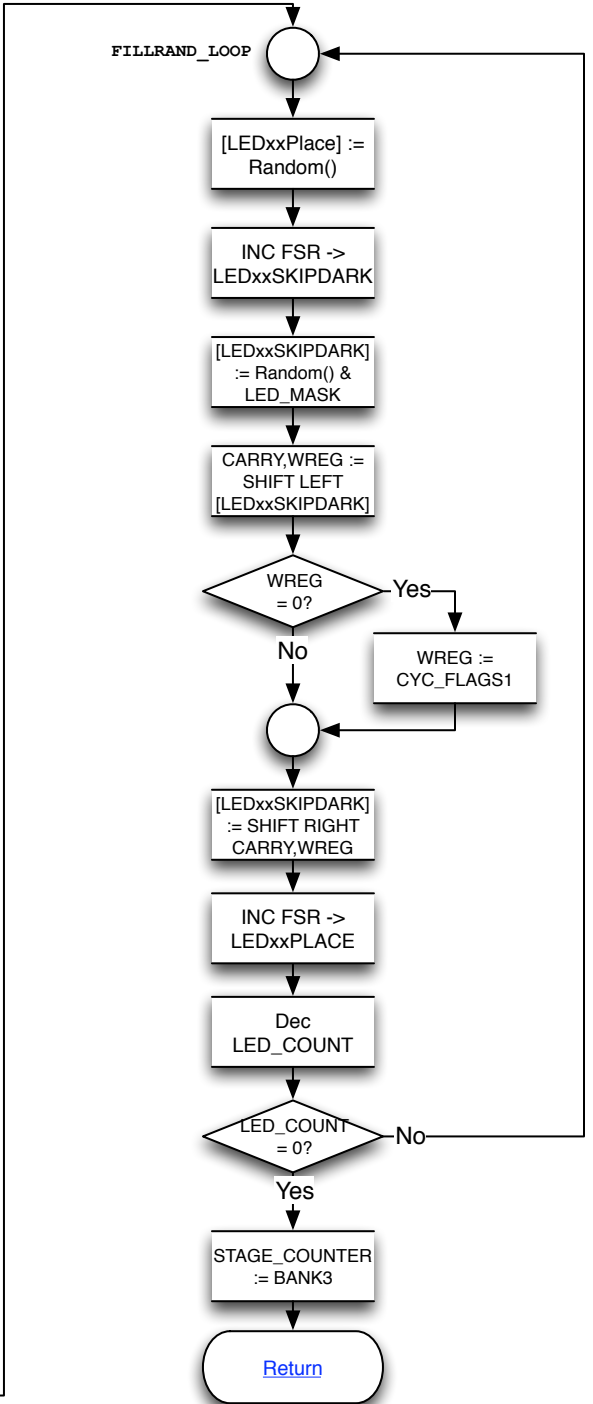
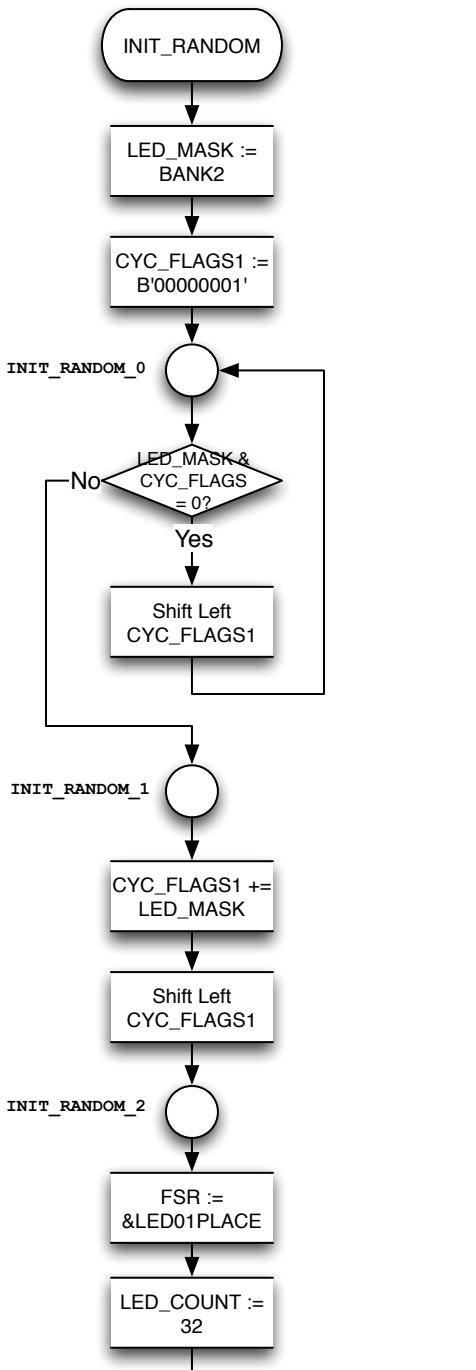
[Init Drip](#)

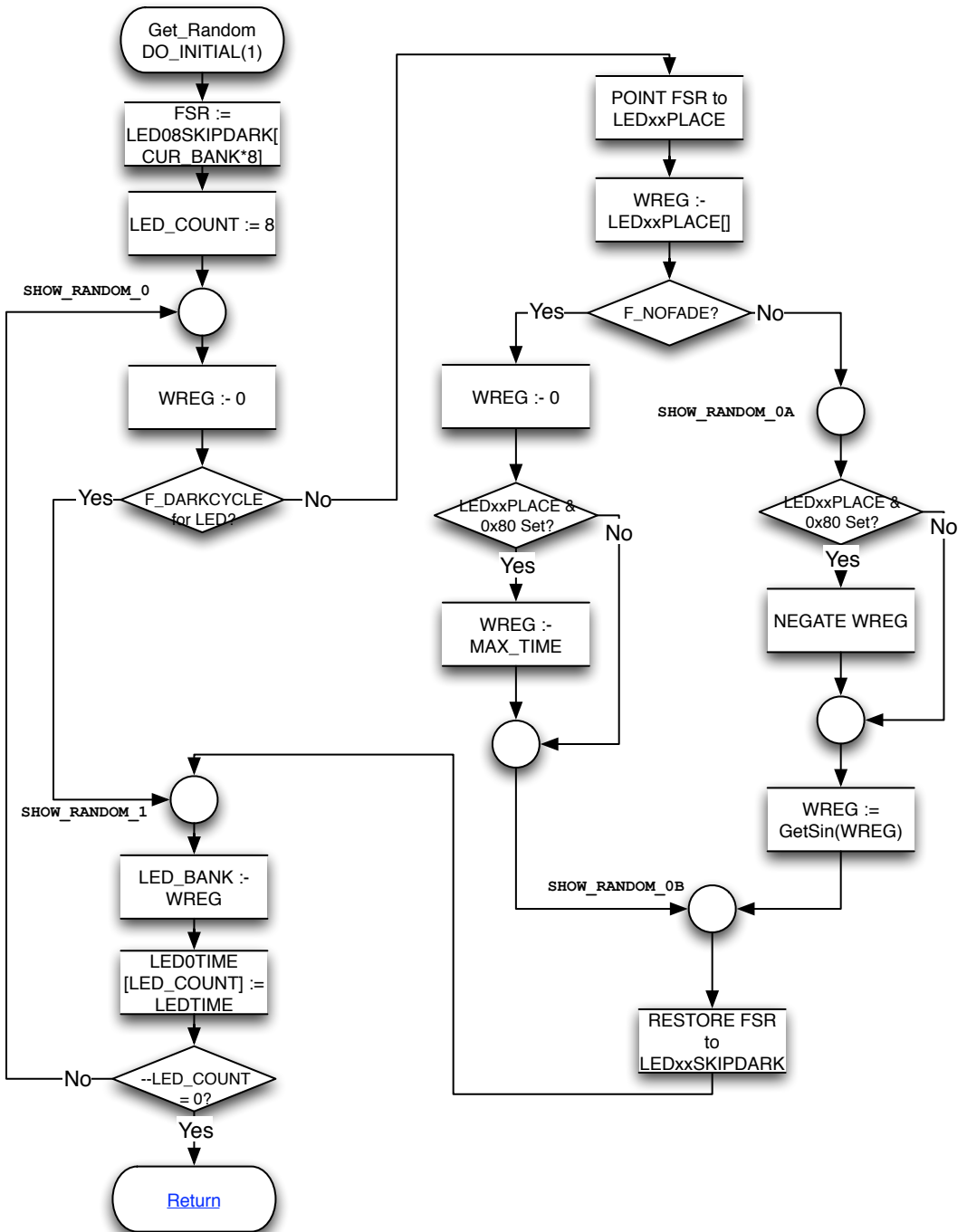
[Refresh Drip](#)

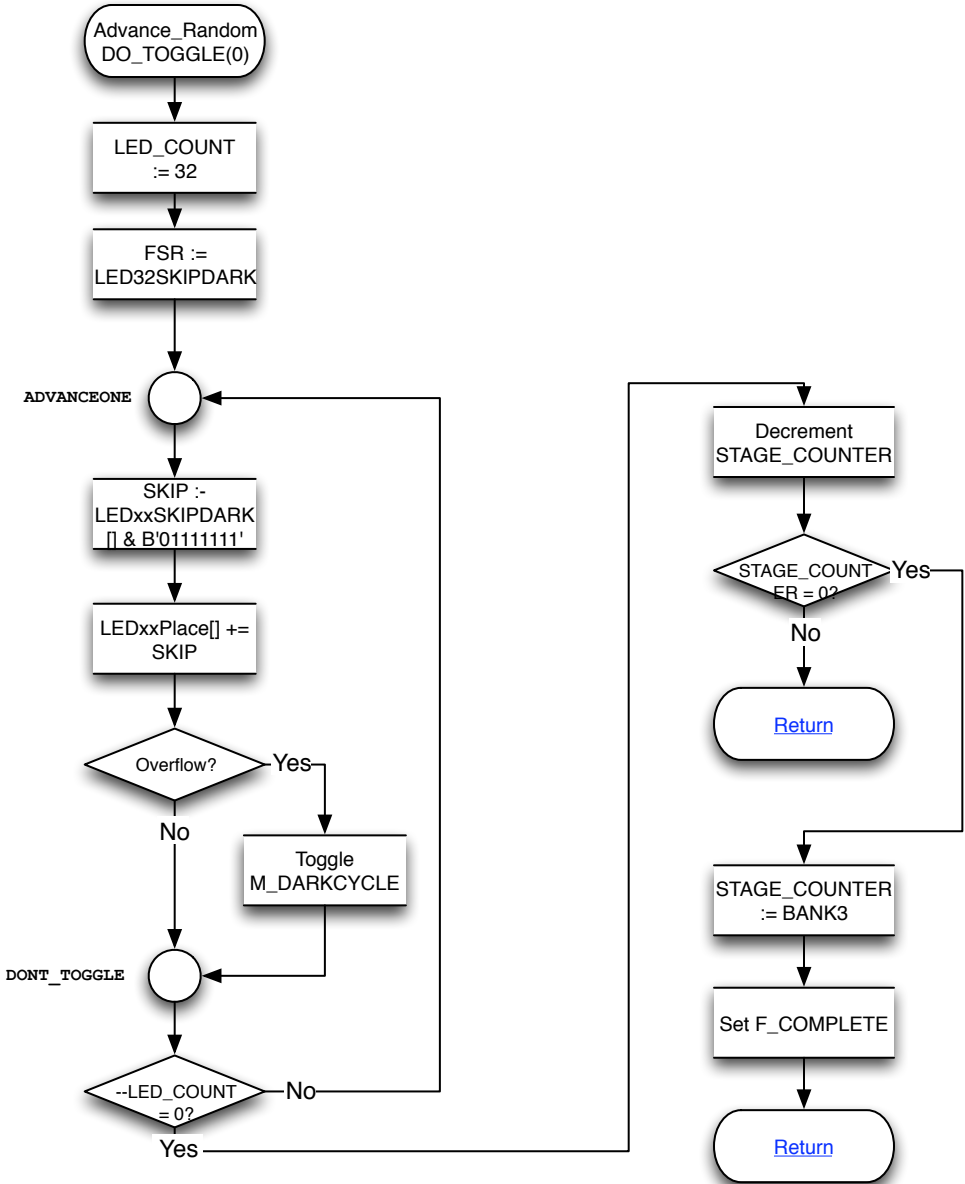


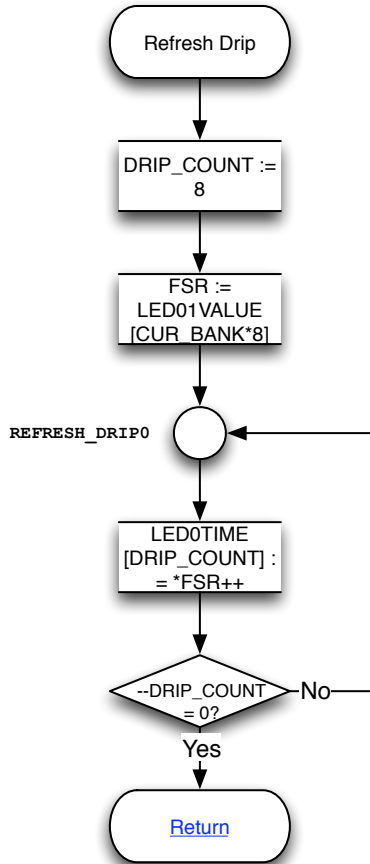
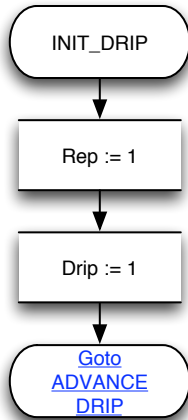
Function Advance Routines
[Advance Random](#)
[Advance Drip](#)

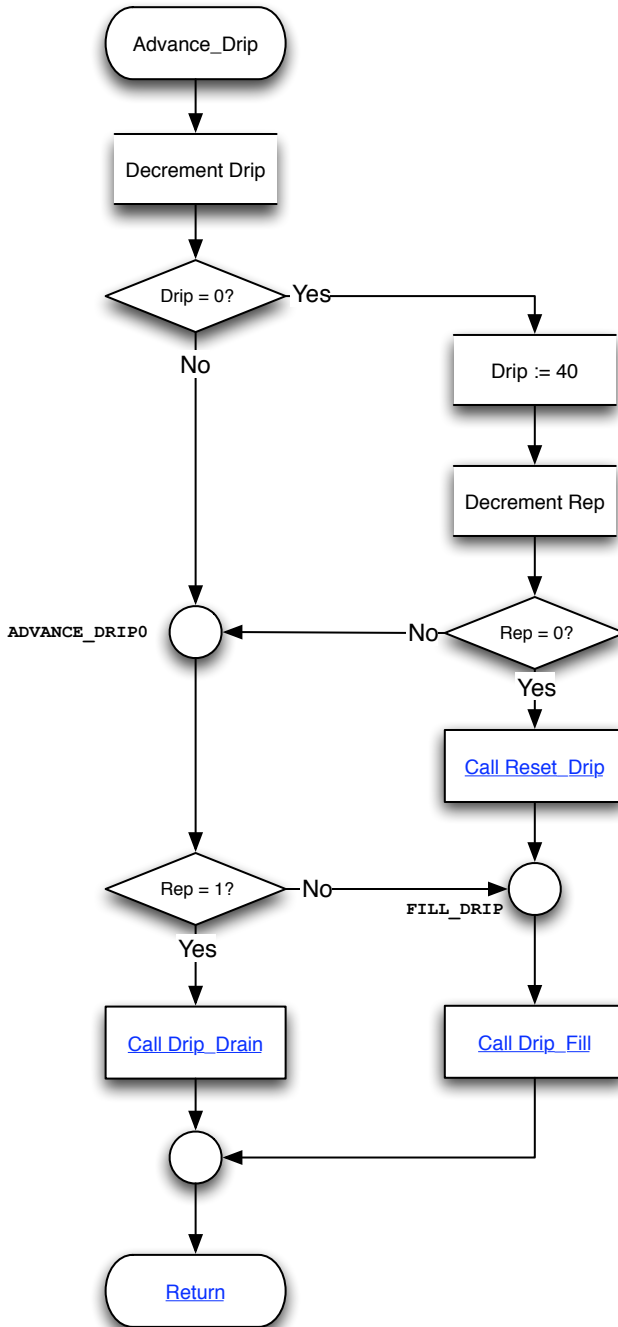
Init Random

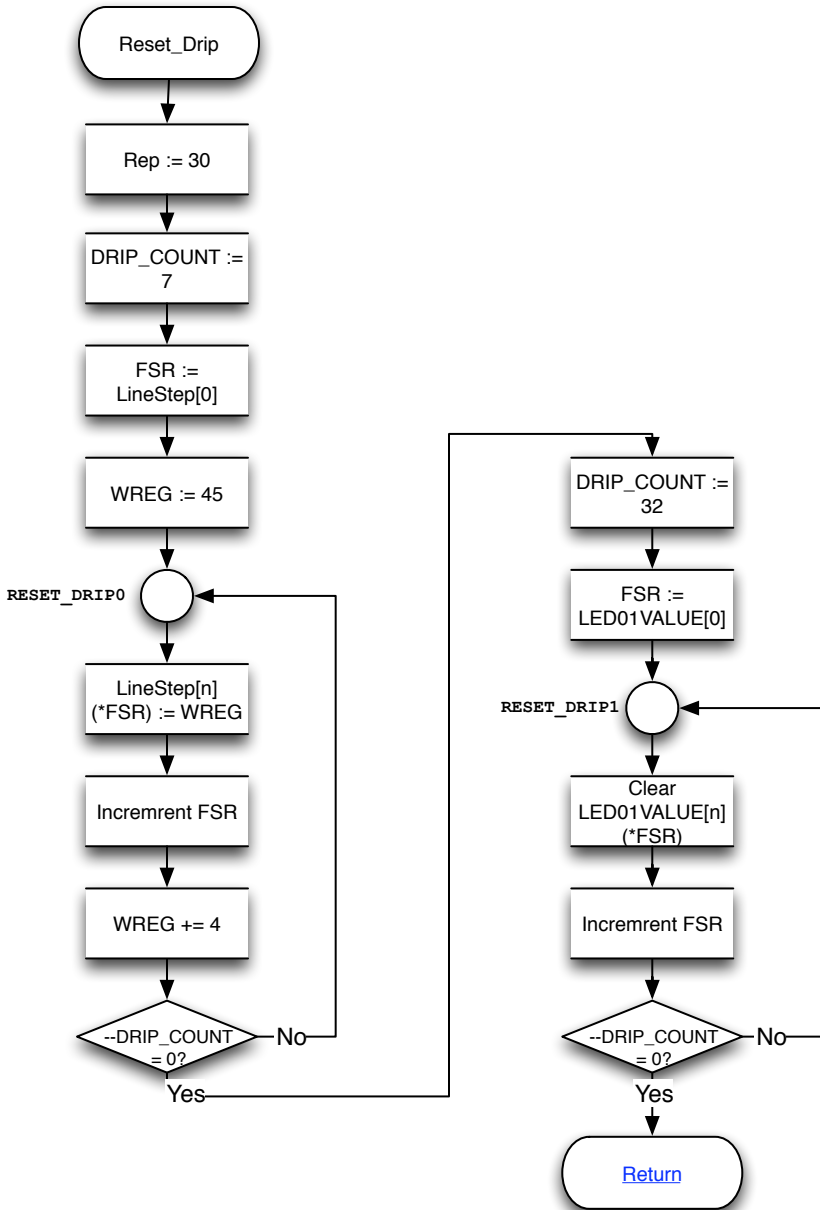


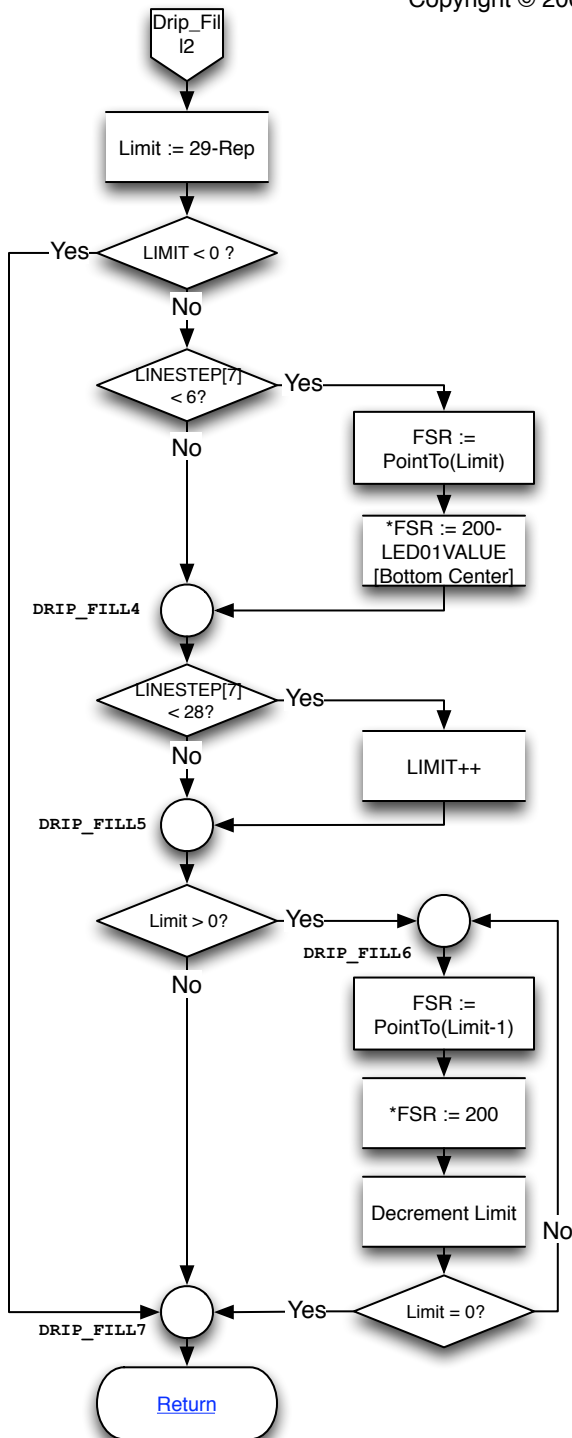












Drip Drain

