EEE4084F Lecture 20 Class activity

Implement a pattern counter as follows:

- Send a starting address (byte addressing used)
- Send an ending address
- Send a 2-byte (16-bit) sequence to search for (this can be a 16-bit input)
- Send an enable pulse to read all the inputs and start searching (note the inputs may change later during the execution)
- Returns a 16-bit integer showing the number found, and raises done to high
- Assume that the pattern counter has an internal 2Kbytes of memory called mem that it will be search through to find the pattern (don't worry about how this memory is set up).

The diagram below shows a how you could visualize the pattern counter device.

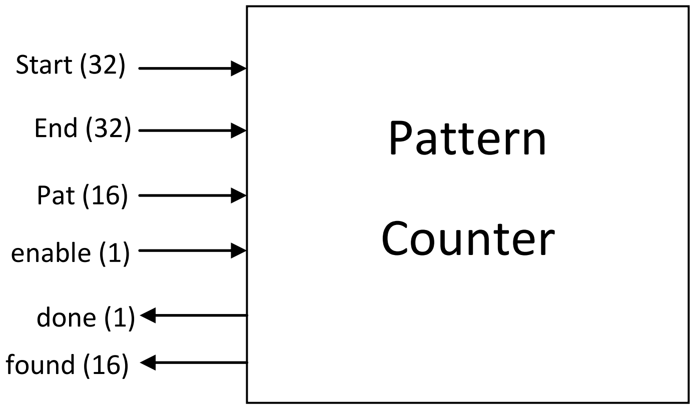
Operation of the pattern counter

Make your counter implement the following operations that are expressed in pesudocode:

```
• Unsigned at;
• Unsigned short pat;
• byte mem [2048];
• Whenever enable is changes from 0 to 1 do the following:
    o at = input start; found = 0; done = 0;
    o check that at and start and within the 2K bounds, exit if out of bounds
    o while (at < end) do // i.e. do nothing if at=end as we need min 2 bytes
        ▪ if ((unsigned short)mem[at] +256 *mem[at+1] ) == pat then
            • set found = 1, set done = 1; stop the operation.
        ▪ else
            • at = at + 1
    o done = 1;
```

Your tasks:

1. Implement the pattern counter as a standard C function
2. Concert your standard C function into a form that is closer to a form that can be converted into HDL (i.e., you need a means to buffer the input, activation and done logic, etc.)


Start (32)
End (32)
Pat (16)
enable (1)
done (1)
found (16)
Pattern
Counter