Circuit design for LED monitoring system

By

William Lash

Abstract

The purpose of this project was to design a Led circuit that would contain a large amount of LEDs, activate them by address, then holds the flashing addressed Led in memory and activates another Led to blink, allowing the circuit to have multiple flashing LEDs, in the final circuit there will be around a thousand LEDs. This massive circuit board will then be use to run experiments on about a thousand crystal samples of Lead Tungsten at the Jefferson Lab.

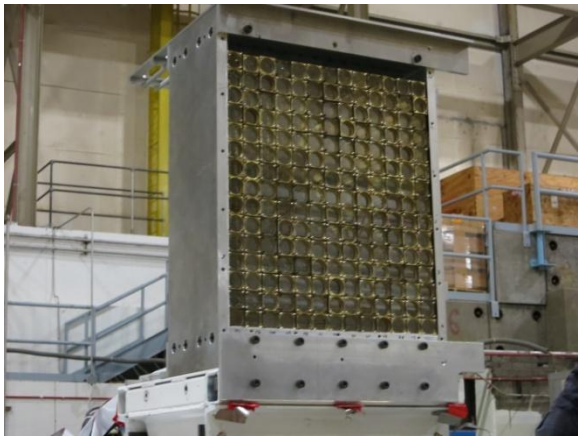
We first look for the components that could perform these actions, so we looked at a similar circuit board at Jefferson lab to get a general idea of what we needed to look for. We also did some online research to look for components and to find alternative methods of building the circuit. We found several alternatives to making a circuit by hand, but none of them met the requirements. So, we decided to go on ahead and build the circuit based on the ideas we came up with from examining the similar circuit at Jefferson lab.

Our research told us, we only needed three different components to make a circuit that would be able to address different LEDS, then remember their last state, and be able to change the frequency at which the addressed Led Would blink. Those three components are a two-input AND gate, knot or inverter, and a D flip-flop. We used 7404 also known as a hex converter, two 7408 which contains four two-input AND gates each, and two 7474 which contains two D flip-flops each. With which we were able to hook up four LEDs that were addressable, had memory, and were able to change frequency.

In conclusion with three simple components we were able to create a circuit that can be attached to a controller. Then programmed to address the LEDs have them blink then control which LED is flashing and at what frequency. In the future I hope to make the program for this circuit more user friendly, so that it's easy for the researchers to use.

Introduction

The goal of this project was to design a Led circuit that would contain many LEDs, activate them by address, then holds the flashing addressed Led in memory and activates another Led to blink, allowing the circuit to have multiple flashing LEDs, in the final circuit there will be around a thousand LEDs. This massive circuit board will be for a calorimeter, that will be use to run experiments on about a thousand crystal samples of lead tungsten at the Jefferson Lab. The circuit will be used to cure the crystals, when exposed to radiation the crystals transparency changes this cause interference with some of the experiments. So to change them back you must expose them to low level photons or visible light provided by the LEDs, so our circuit is setup to provide control over the LEDs used to cure the crystals.



Definition of a circuit

A circuit is a flow of electrons provided by a voltage source; this voltage then passes through various components such as resistors, transistors, LEDs, and many others; into a return or ground. A very basic language that a circuit uses to perform its functions is known as binary. In binary voltage and ground are labeled as 1(voltage) and 0 (ground). Each circuit has its own function or purpose, for example one circuit might simply light up a LED, while another may compare two input signals (voltages) and produce an output signal. In our circuit we used integrated circuits to perform various functions. An integrated circuit is a circuit contained on a single small "chip" of semi conductive material, which performs a specific function. We use three integrated circuits in our prototype, along with four LEDs and resistors; this is excluding the controller of course.

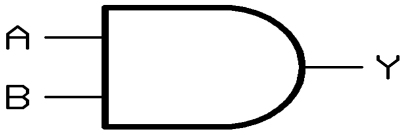
Definition of components

Every Circuit contains different components to perform certain tasks. These Components range from simple, like a resistor or capacitor to very complex, such as integrated circuits or micro processors. Components can create memory or setup addresses for other components and also changes the signal of a circuit. In this circuit we have 5 components to do different tasks.

The Components

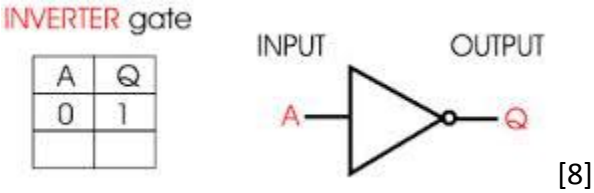
We used 5 different components 4 LEDs, 4 470 Ohm resistors, 4 two-input AND gates, 1 hex inverter, and 4 D flip-flops. All contained in 5 integrated circuits, except the LEDs and resistors.

A two-input AND gate takes two inputs of binary signal 0 or 1, and produces an output based on them. If the inputs match like (1, 1), it produces an output of (1) or if the inputs are (0, 0), the output is (0). If the inputs are different the output will be (0).This can be used in a variety of ways, such as setting up addresses, switches, and blinking. We used a 7408 two-input AND gate Integrated circuit, which contains four AND gates.



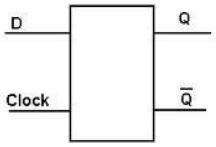
[9]

The 7404 hex inverter contains six inverters, also known as knots that invert binary signals. For example, if you send a binary signal of zero to the input it will out put a binary signal of one and vice versa. We use this in the addressing system of the circuit, so that all the addresses produce a voltage or a binary signal of one, instead of a binary signal of 0, so that every address will light up the LED.



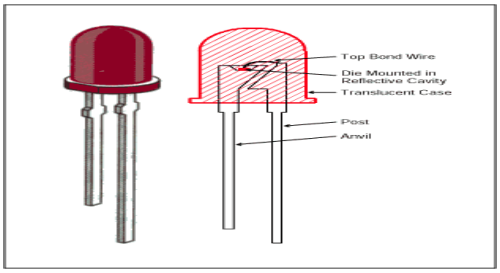
[8]

[4,5] A Flip-Flop is a circuit that has two stable states and can be used to store state information. The 7474 is an integrated circuit that contains two type D flip-flops. The type D flip-flop can be made to change state by signals applied to one or more control inputs and will have one or two outputs, with this we give each LED memory.



[10]

The LEDs and resistors are in a series circuit. LEDs or light emitting diodes are a component that emits light when voltage is passes between two leads. The resistor are used to decrease voltage and create current, in our circuit their used to shield the LEDs from burning out from the applied voltage. We are using all the other components to manipulate the LEDs to blink at a given frequency, have memory, and be addressable by our controller.



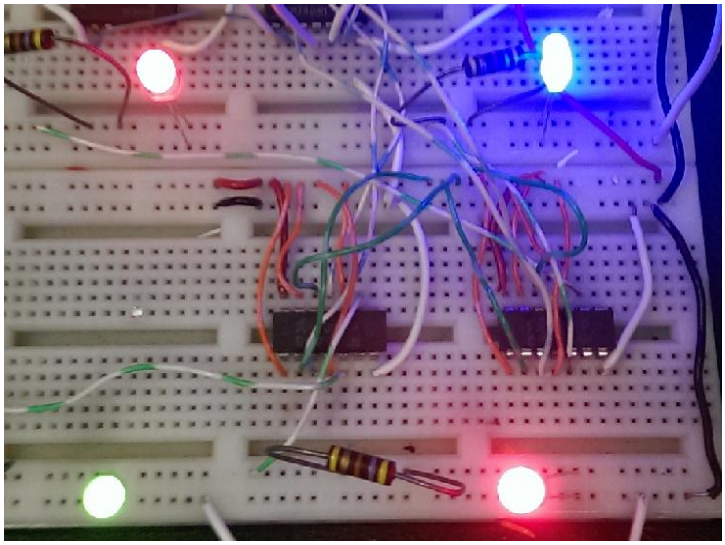
[11]

First prototype

The first prototype was made of four 7408s to make 4 addressable LEDs. We had the prototype setup to take binary inputs of 1 or 0 and based on those inputs a certain LED would light up. But any addresses that contain a binary signal of 0 wouldn't light up when addressed. We found the reason was that the AND gates only produce a binary signal of 1 when both inputs are also a signal of 1. So we had to come up with a way to convert all the binary signals of 0 to 1, while still being able to address each LED. Our solution was a hex inverter.

Addressability

To make the LEDs Addressable we combined the hex inverter with the 7408 two-input AND gate. The AND gate requires two binary signals of one to create a binary signal of one output, so when we tried to access the LED with the binary signal (0,0) or (0,1) or (1,0) it wouldn't turn on. To fix this we ran all the binary signals of zero through the inverter to create binary signals of one, thus turning on the LED, but only if the binary signal send was zero or else the LED would remain off. Now we could address our four LEDs individually using this new circuit. The addresses are form in a four bit system, so there is four possible addresses, address (0,0), address (0,1), address (1,0), and address (1,1). Each is assigned to a different LED and so when we send two binary signals of either 0 or 1; it will address the given address. For example, we send a binary signal of 0 and 1, the LED with the address (0,1) will light up.



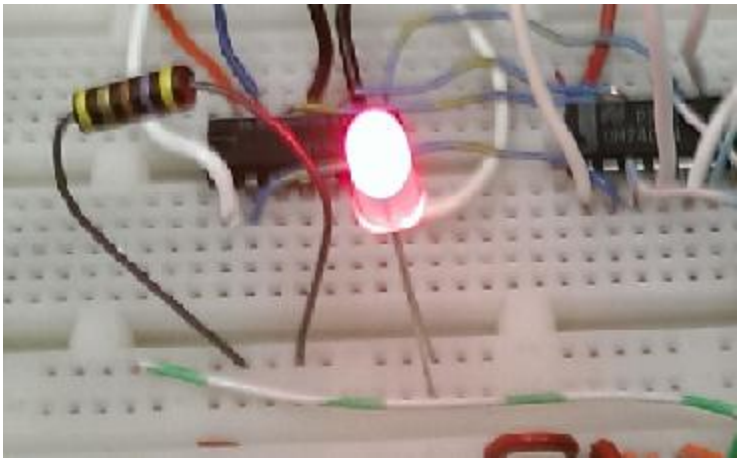
Memory

[1]Memory circuits function by storing the voltage present on an input signal whenever they are triggered by a control signal, and they retain that stored voltage until the next

assertion of the control (or trigger) signal. Between assertions of the control signal, the input signal is ignored and the output is driven to the most recently stored voltage.

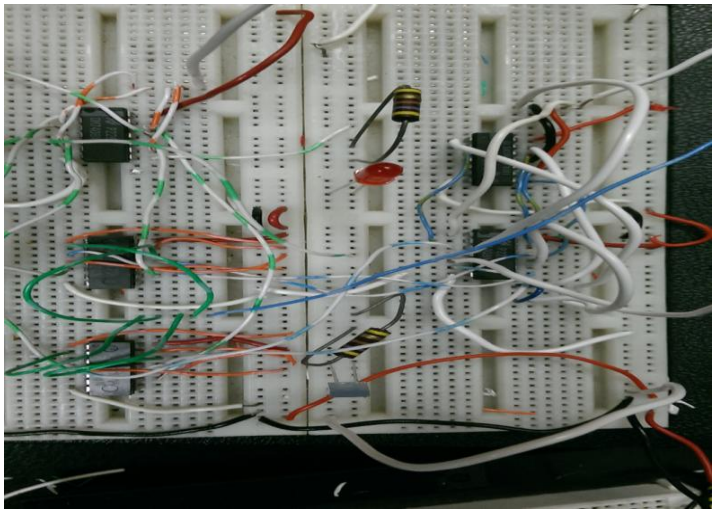
Since a memory circuit stores the input signal level at each assertion of the control input, the output will change immediately after the control signal is asserted (if the input value is opposite to what is stored), or it will remain constant. “Memory” occurs between control signal assertions, because the output remains constant at the last stored value, regardless of input signal changes.

[4,5] In a D Flip-flop memory is stored by being able to change the LEDs state and remembering the last active state. Each time we address a Led we send a binary signal of one or zero to the D flip-flops Data input (D) which determines, if the Led will turn off, on or remain at its current state. When we want to change the state of different LEDs we first address the Led we want to change and then send a binary signal of 1 or 0 to the data input, which determines the state of the led. If we send a signal of 1 the addressed LED turns on, if we send a signal of 0 the addressed LED turns off, all other LEDs are unaffected. The addressed Led stays in the last known state until addressed once more. In this way our circuit has memory for each LED.



Frequency adjuster

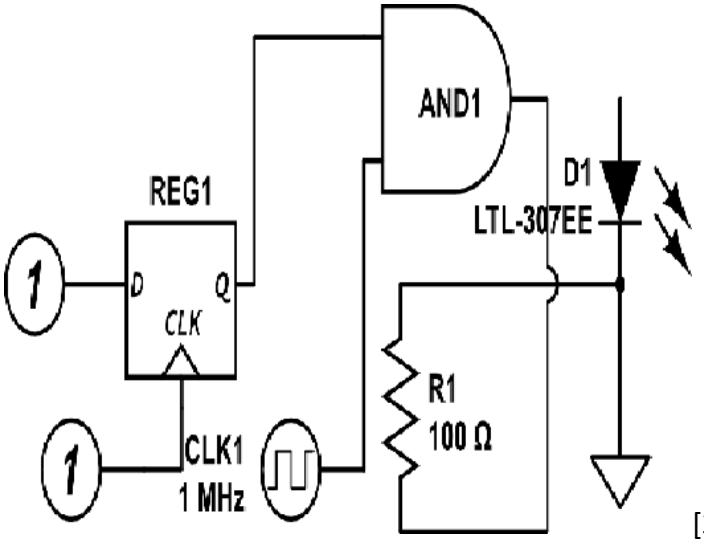
One of the functions we wanted for our circuit was a way to adjust the frequency of the LEDs. We look at several components and realized that a AND gate to do the job. So to adjust the frequency at which the LEDs blink, we simply used 7408 two-input AND gate.[7] We ran the output from the D flip-flop feed into one input of a AND gate, while the other input of the AND gate was hooked up to the controller, which is programmed to send a charge every few milliseconds causing the LEDs that were on to blink. Using the formula [2] (f=1/T) we are able to calculate the frequency at which the LEDs will blink. Using the 7408 two-input AND gate and a program I found on the internet for the controller I was able to make a rudimentary frequency adjuster.



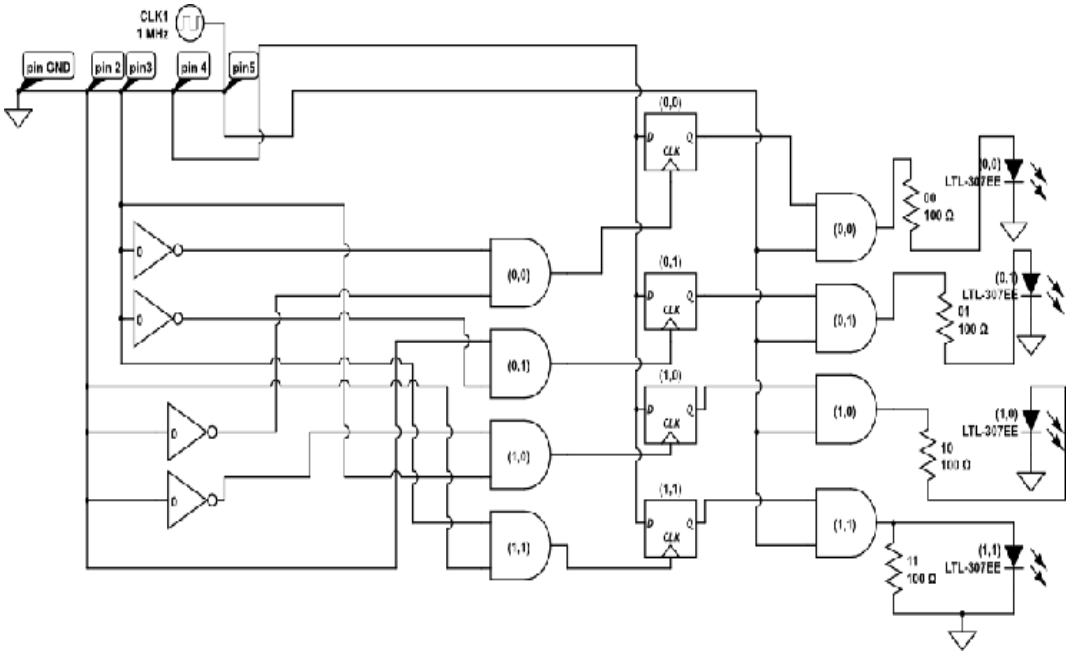
Final product

After all the research and prototyping we have created the final prototype. This circuit has addressable LEDs and each led has memory, so we can turn more than one on at a time and have them blink at adjustable frequencies. These are all functions we set out to include in the circuit and so I believe this prototype is ready for the next stage.

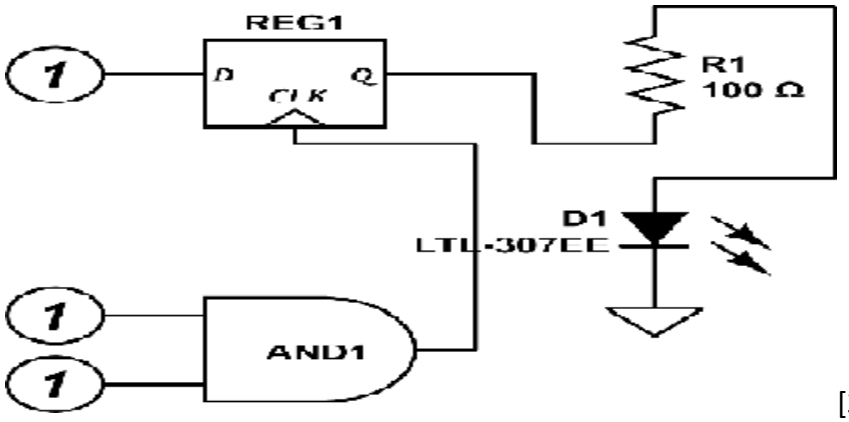
Circuit diagrams



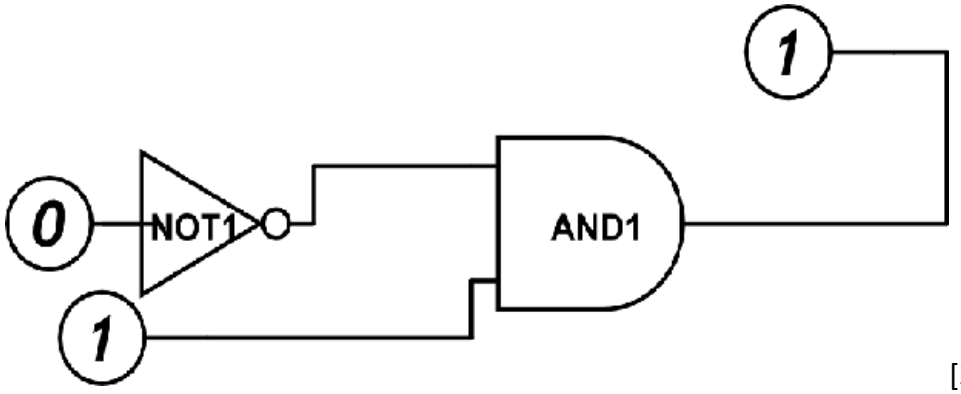
[3] memory



[3] full circuit



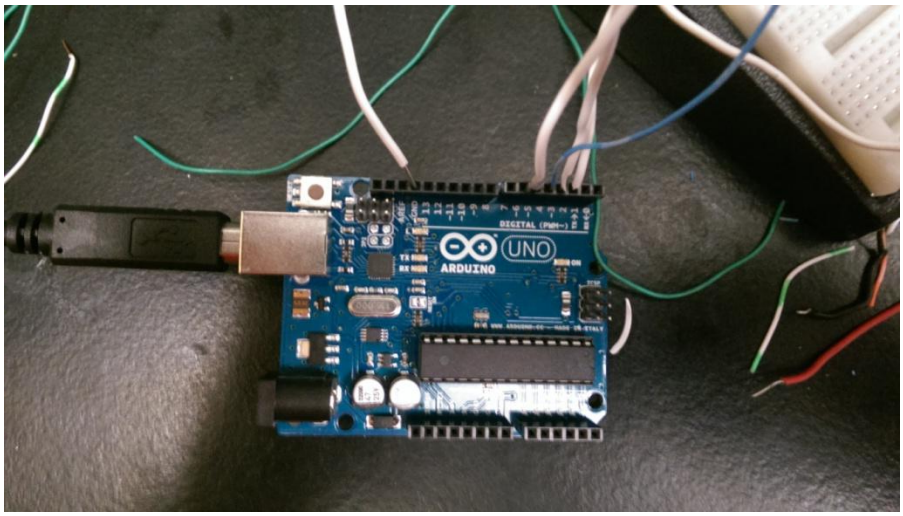
[3] frequency adjuster



[3] addressing

Adriuno Uno

[6] For this project we chose an Arduino Uno to be our controller and have programmed it to test the addressability of the LEDS. Also, we are working on making sub programs to do different tasks, such as all off, all on, and two at a time. In the future I hope to develop a sequence so that the LEDs light up in a given order or on command and turn off in a different order, so that the researchers can remotely turn on and off different LEDS. The Arduino Uno uses the simple c programming language. Most of the programs we used can be found online but more complex ones will take time to produce from the simple examples online.



Codes

LED light up

```
void setup()
{
   pinMode(4,OUTPUT);
  pinMode(2,OUTPUT);
  pinMode(3,OUTPUT);
  pinMode(5,OUTPUT);
}
void loop()
{
digitalWrite(5,LOW);
int led1 = HIGH;//01
int led3 = LOW;//11
int led2 = LOW;//10
```

```
int led0 = HIGH;//oo

digitalWrite(4,led1);


digitalWrite(2,LOW); // turn on digital pin 2

digitalWrite(3,HIGH ); // turn on digital pin 2

digitalWrite(4,led3);

digitalWrite(2,HIGH); // turn on digital pin 2

digitalWrite(3,HIGH);

digitalWrite(4,led2);

digitalWrite(2,HIGH); // turn on digital pin 2

digitalWrite(3,LOW);

digitalWrite(4,led0);

digitalWrite(2,LOW); // turn on digital pin 2

digitalWrite(3,LOW );

}
```

LED Blink

```
oid setup()

{

   pinMode(4,OUTPUT);

  pinMode(2,OUTPUT);

  pinMode(3,OUTPUT);

  pinMode(5,OUTPUT);
```

```
}

void loop()

{

int led1 = HIGH;//01

int led3 = HIGH;//11

int led2 = HIGH;//10

int led0 = HIGH;//oo

digitalWrite(4,led1);

digitalWrite(2,LOW); // turn on digital pin 2

digitalWrite(3,HIGH ); // turn on digital pin 2

digitalWrite(4,led3);

digitalWrite(2,HIGH); // turn on digital pin 2

digitalWrite(3,HIGH);

digitalWrite(4,led2);

digitalWrite(2,HIGH); // turn on digital pin 2

digitalWrite(3,LOW);

digitalWrite(4,led0);

digitalWrite(2,LOW); // turn on digital pin 2

digitalWrite(3,LOW );

const int ledPin = 5;

int ledState = LOW;
```

```
long previousMillis = 0;
long interval = 1000;
unsigned long currentMillis = millis();
if(currentMillis - previousMillis > interval){
  previousMillis = currentMillis;
  if (ledState == LOW)
  ledState = HIGH;
  else
  ledState = LOW;
  digitalWrite(ledPin,ledState);
}
}
```

Conclusion

At the completion of our project we have produced a circuit that addresses LEDs that are able to remember being addressed until readdressed and told otherwise, so that we can access each LED individually and control the frequency of they're blinks. This is all done through our controller an Arduino Uno, Which we programmed to perform the required functions. After we have sorted out all the bugs in the programs, we must make a full size circuit diagram for the 1000 Led board. Our prototype does not scale up very well so we must find different components that can handle the large amount of LEDs we must control. We think we need to create a ten bit circuit to support the calorimeter, but more research is required to get to this stage. After we make a better large scale circuit, we will send the

diagram to a circuit board manufacturer, so they can make the huge board for the calorimeter that will be used to cure the crystals of their defects from being exposed to radiation, so that there is no interference with the experiments at Jefferson Lab.

References

1. https://www.digilentinc.com/classroom/realdigital/M9/RealDigital_Module_9.pdf
2. https://en.wikipedia.org/wiki/Frequency
3. https://www.circuitlab.com/editor/
4. http://www.nxp.com/documents/data_sheet/74HC_HCT74.pdf
5. http://www.ti.com/lit/ds/symlink/sn54s74.pdf
6. http://playground.arduino.cc/Main/LedControl
7. http://arduino.cc/en/Tutorial/BlinkWithoutDelay?action=sourceblock&num=1
8. http://www.technologystudent.com/elec1/dig3.htm
9. http://eprojex.net/php/I/G/a.php?id=191
10. http://www.hobbyprojects.com/flip_flop/clocked_D_type_flip-flop.html
11. http://www.maximintegrated.com/en/app-notes/index.mvp/id/1883