

The Original Vision:

So I have a fat cat and a skinny cat. Invariably, the fat cat eats some of the skinny cats food. Both of the cats have an RFID tag in their necks. What I want to do is basically scan my cats when they go to the food bowl. Based on what cat it is, only a certain food bowl will open.

Project Reality:

The following items were purchased or cost the following for my project:

- CAT MATE C10 \$23.98
- RFID Reader Kit With Serial Interface \$184.00
- 9v Power Adapter
(9V/800mA AC-to-DC Power Adapter)
for Serial Reader \$19.99
- Futaba S3001 Servo Motor \$19.99



Figure 1: CAT MATE C10



Figure 2: RF ID DEMOKIT-1 Intersoft Corp



Figure 3: Z8 Encore



Figure 4: Futaba S3001 Servo

Changes:

My cats have chips that need to be read at 125k with the amplitude modulation of "FSK." This is the modulation that the homeagainid.com pet chips support. They support no other modulation. This is listed in detail in the ISO 11785 Annex A 2.1. While the homeagainid.com transponders (chips for pets) are not proprietary, from the searching I did, you would need to create your own RFID reader capable of reading the FSK modulation. To the best of my knowledge, there is no product openly available for reading the transponders they sell. I must stress that they are using an open standard and it is not proprietary. The information on the chip is also not encrypted according to the homeagainid.com technical representatives.

In my proposal submission, I did not have a motor included. I have since bought two servos. One did not work because it did not operate in multiple directions. The last servo by Futaba works.

Hardware:

RFID Reader:

- Needs a 9v power supply (Zilog Board could provide power)
- Enclosure contains female DB9 serial interface and RFID reader components (antenna and RFID reader). You must use a NULL modem to connect to a standard male to female DB9 cable between the board and the reader.
- The RS232 transmission contains no line feeds and will constantly send either " " or an RFID tag.
 - RS232 operates at 9600 baud with no hardware flow control.
- Tags will be given in the following format from the RFID Reader via the RS232 interface " :#####" ..
- The RFID Reader comes with a variety of RFID tags that all have a length of 13 characters.

Futaba S3001 Servo:

- This is a standard servo with three wires
 - Ground
 - Power (using the 5v from the Zilog)
 - GPIO
- This Servo operates between 4.8v and 6v. In my project, I am powering it off of the vcc or 5v pins.
- Torque between 33.3oz and 41.6oz per inch
- Size 1.6x.8x1.4 inches

CAT MATE C10 (not implemented):

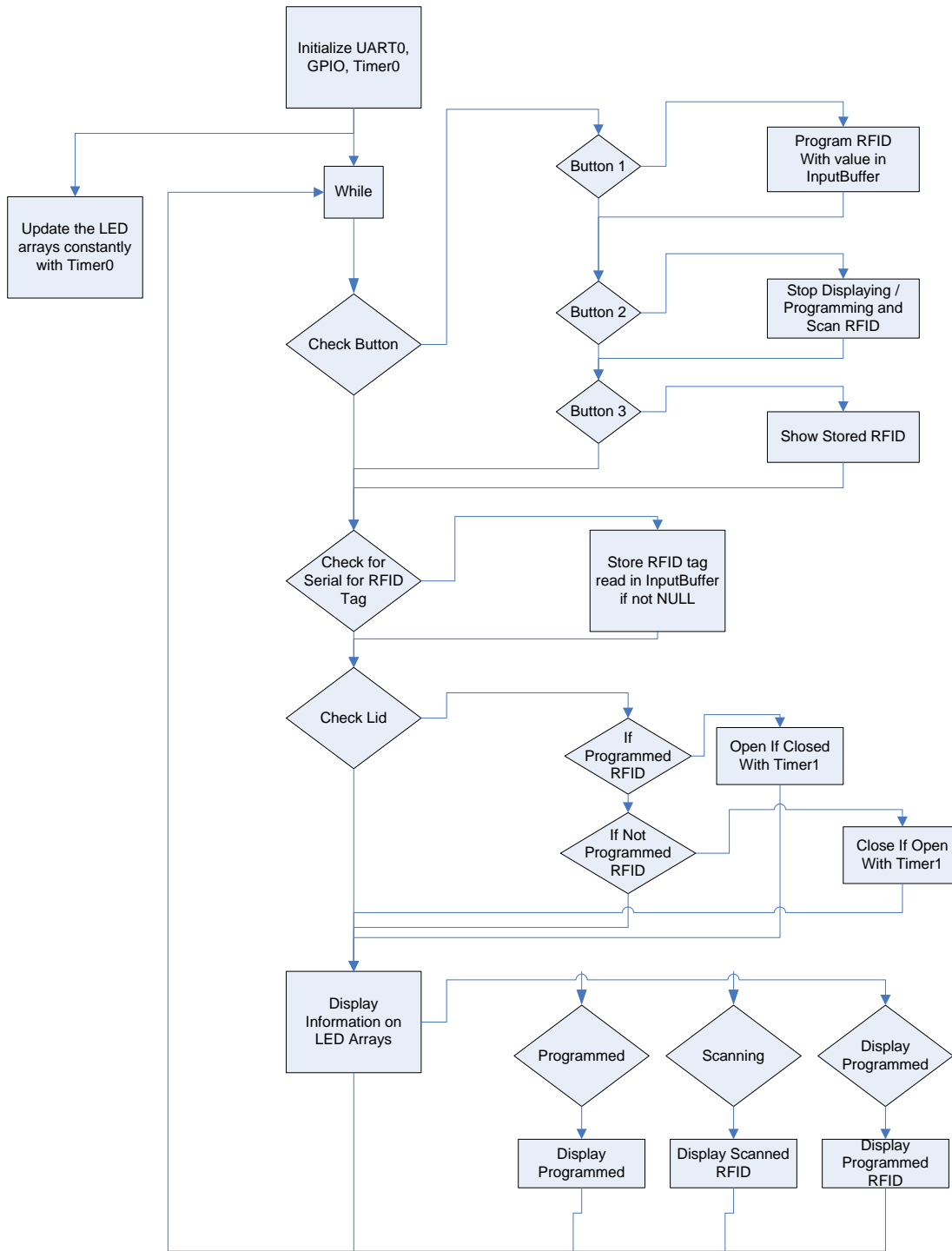
- This is a small plastic pet feeder that can be completely dismantled.
- It is the following components
 - Lid (can be removed from unit)
 - Bowl
 - Timer (AA powered)

- Spring to open lid when timer reaches release point

Zilog Z8 Encore:

- There are many components to the Z8 Encore. For more information on the variety of abilities that it provides, please refer to the Technical Reference Manual located at <http://www.zilog.com/docs/z8encore/ps0176.pdf>.
- I am using the following features of the Zilog Z8 in my project
 - All 4 LED arrays
 - The RS232 (Console) connection for reading the RFID Reader
 - All 3 buttons
 - SW1 for Programming an RFID tag.
 - SW2 for switching to scan mode.
 - SW3 for displaying the programmed RFID tag.
 - I am using the 5v power on the board (vcc pin) to power the servo motor
 - I am using the ground for the board for the ground of the servo
 - I could also use the 9v power and ground for powering the RFID Reader, but opted to use a power adapter that supports its power plug port.
 - I am using two timers on the Zilog board.
 - Timer0 is used to update the display information on the 4 LED arrays
 - Programmed RFID
 - RFID Scanned
 - Timer1 is used for Pulse Width Modulation (PWM) to control the servo.
 - The servo is connected to the PC1/T1OUT port for GPIO.

Software Block Diagram:



From the above flow chart, I am doing the following:

The first thing that is done in the software is all components are initialized. This includes all of the buttons, timer0, uart0, and the LED arrays. We then go into an infinite loop that runs from power on to power off of the Zilog board. In this while loop, the buttons are checked for user input, the serial interface is checked for an RFID tag, and the lid is checked to make sure it is in the correct state. The lid

state is determined on the current position of the lid (open or closed) and the comparison of the programmed RFID tag and the read RFID tag. Currently, only a single RFID tag can be programmed (which works out well as the feeder only feeds a single cat).

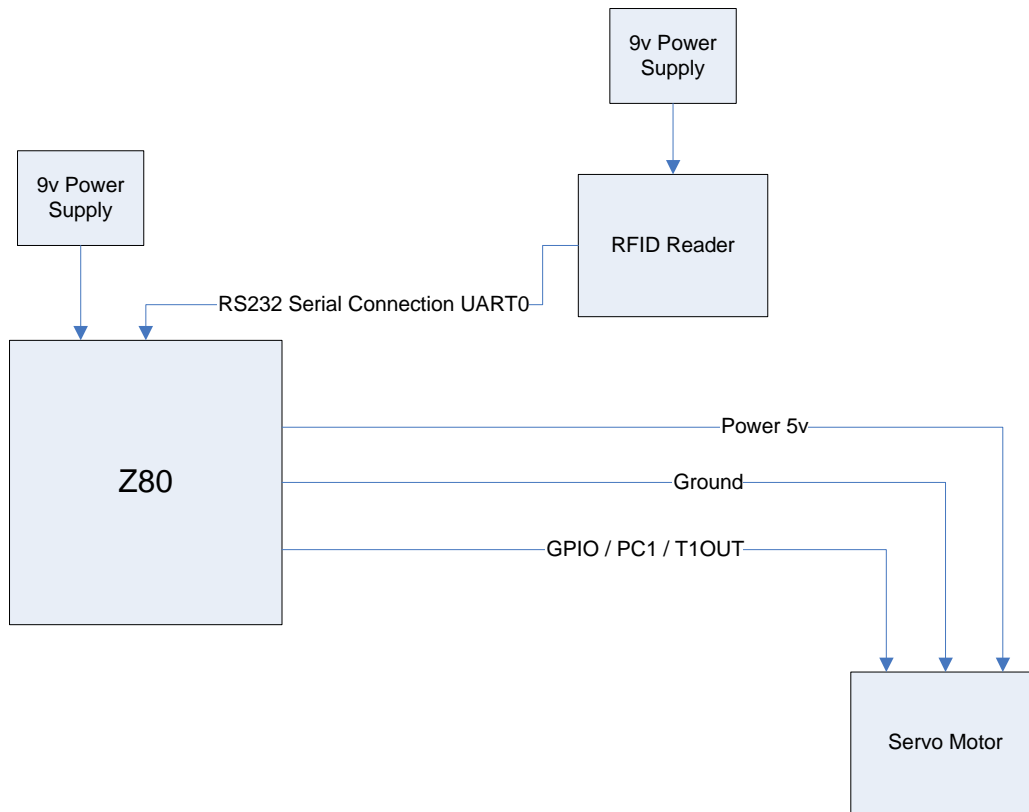
The buttons are checked before we read the serial interface because they must read the active RFID tag. Otherwise, it could be possible to accidentally switch a tag between button push and storage of the read tag. The First button will actually tell the board to store the active RFID tag as the RFID tag that is allowed to eat food from the bowl. Button number two is used to switch back from programming or display-programmed mode into an active scan mode. The third button just shows what has been programmed on the board without actually programming an RFID tag.

In the Check Serial for RFID function, we are constantly scanning for a new RFID tag. If a null tag is read, it is skipped. If we are currently showing the programmed RFID tag, we do not read any tag. When a new RFID tag is read, we store that tag into an input buffer that is checked against and store in the programmed RFID tag.

In the Check Lid function, we are testing the stored RFID tag against the read RFID tag. If we need to open or closed the lid, we will enable timer1 with PWM to close or open the lid (clockwise or counter-clockwise on the servo).

Timer0 is being used to update the information being displayed on the LED arrays. Information is displayed by printing a character a single row at a time across all the LED arrays from a print string.

Hardware Block Diagram:



General Information:

All devices used must be plugged into the Zilog board and powered before powering the Zilog board (if not powered by the Zilog board).

Connections:

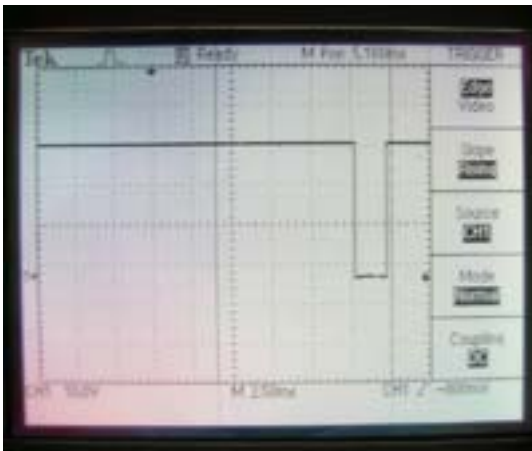
- When connected the RFID Reader with a standard male-to-female DB9 cable, it is important to use a Null Modem or you will not get any data via RS232.
- RFID Reader can be powered via the 9v on the Z8 Encore board if you want to wire it directly, but you will need to buy a male 9v power plug. I chose instead to use a separate 9v power adapter for the reader.
- The servo is connected in the following way:
 - Green goes to T1OUT or PC1
 - Black goes to ground
 - Red goes to vcc or 5v

This is a standard compile with the “Zilog ZDS II - Z8Encore! 4.9.1” IDE. Uart0 must be enabled.

With the servo, I am sending it a 1ms high pulse over 14ms period to open the

lid. The 1ms pulse is high, and the polarity is low. To close the lid, I set the PWM back to the start state. This is not perfect as it is not idle when it gets to this position. To be honest, it is a hack I did and it seems to work. I could change the code to close the lid and then just disable the timer to make the lid close, but I have not done that.

I actually brought an oscilloscope home over a single night to track the waveforms that were being sent from the GPIO pin T1OUT to see what was happening with my setup. It was the only way to fully debug my code not working. By learning to use the tool and then see what kind of pulse my code was producing, I was able to see that I needed to switch the polarity. Once that was done, my servo began to function more correctly. Below is a picture that shows the polarity being high.



Encountered Problems:

They were numerous.

The built-in Zilog API for reading from UART devices only support reading from a default UART interface (UART0). So, you are not able to use two UART devices without rewriting a large portion of the Zilog code, or implementing your own features. In addition, the Zilog API for reading from UART0 will not transmit information until a carriage return or line feed has been entered. A quick fix to both of these issues for me was to use a single UART (UART0), and to use "scanf", which does not wait for a carriage return.

I initially bought the wrong servo. It was a two-wire servo (ground and power) and it operated at the wrong voltage. While I was able to downgrade the power (9v) coming from the z8 board to power the servo, it would only turn in a single direction. So, I was unable to open and then close the lid (or turn clockwise and counter-clockwise the servo for the lid). This then necessitated a last minute run to a local hobby store (hobbyworks) to purchase a standard three-wire servo that

operated at 5v. This enabled me to forgo using the breadboard and to directly connect the servo to the Zilog board. This was not immediately apparent, however, because I did not read the specifications of the z80 board to see that it output 5 volts on the vcc pins. A result of this was that I at one point was downgrading the 9v from the z80 board with a 5v regulator on a breadboard (which while unnecessary, worked).

In keeping with my failure to read the specifications and schematics of the z80 board, I also picked the wrong pin twice (for the servo GPIO), which caused a wasted day of work. I mistakenly picked an output pin of a button and timer that I was already using in the project. The result was mysterious servo actions and control. After having this pointed out, I was able to correctly select the timer output pin for the timer that I was using. While this was obvious to some, I was not, but now it is crystal clear and I value this new found knowledge.

Incomplete:

I have not attached my servo to the lid of cat feeder. I simply did not have enough time to come up with the best method of attaching the motor to the lid in a reliable way. While I could just slap the motor on the feeder, it would be an eyesore and would have exposed wires.