Week 10 Lab

Two choices for EEPROM –

Use the internal one.
Add an external one.

The on-chip one is not programmed easily on the robot. Instead, we will add a separate one that we can program in the laboratory.

This EPROM chip will directly replace the static RAM chip that is on the robot.

The static RAM should not be swapped with the EPROM until after this week's milestone is complete.

Transitioning to the EPROM

Could we find a 32-kb ROM that drops into the socket with the same pin outs? No.

The standard ROM chips are the 27256, 27128, etc.

27256 is 32k
27128 is 16k
2764 is 8k

There is a scheme for naming. Figure it out.

The 62 series is RAM; the 27 series is ROM.

The 27256 cannot directly drop into the location where the 62256 is located. However, the 27128 only has two different pins that differ *substantially* from the 62256. Thus, rather than using a 32k ROM, we will be using a 16k ROM.

Whenever reading from the ROM, the PGM signal, which is attached to RW.

It is firmware.

Programming the EPROM

Create the program just as you would one that you will send to RAM. Create the S19 file.

The S19 is a file format that puts your binary as a text file so that it goes down the ASCII terminal to the device that is reading it.

S1 records contain actual data that is to be loaded into memory.

When JBUG gets an incorrect checksum, it reports COM error.

Our EPROM programmer allows you to use the S19 format, as well as a number of other file formats including the Intel HEX file format. These were the de facto standards.

Our assembler could create a .HEX file.

Once the program is working in RAM:

```
ORG 0x8000
MAIN: …
…
…

ORG 0xBFFE
MAIN
```