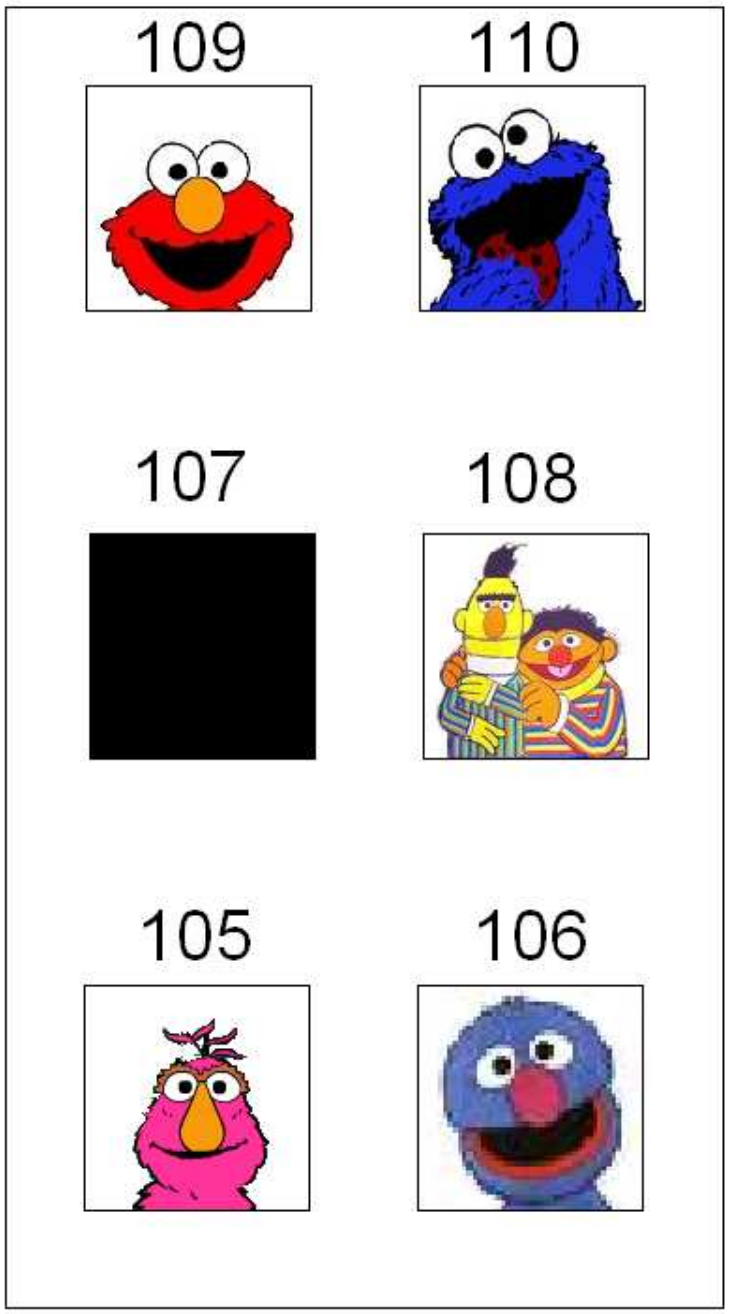
TI-83+ Z80 ASM for the Absolute Beginner

LESSON THREE:

- *On The Street Where You Live*
- *More About RAM*
- *Looking Ahead*

ON THE STREET WHERE YOU LIVE

Let's start this lesson by taking a hypothetical apartment on Sesame Street, where six of the many, many puppets live.



Notice there is no one living in apartment 107.

Suppose that you wanted to visit Cookie Monster. This means you would go to apartment 110 because that is where he is located. But suppose that Cookie Monster is not there. Has his address changed? No, it is still Cookie Monster's apartment. It's just that Cookie Monster is not there, his apartment is empty.

Suppose now that Grover went to visit Cookie Monster. Then apartment 110 is still apartment 110, but its contents have changed. It now contains both Grover and Cookie Monster.

Right now, apartment 107 is empty. But what if Telly decided to move for some reason? Apartment 105 is now empty. The address is still apartment 105, but its contents have changed. Furthermore, the contents of apartment 107 have now changed, as Telly is now living in apartment 107.

EXERCISE:

Suppose that Elmo is hosting a party at his apartment, apartment 109. Telly, Bert, Ernie, Grover and Cookie Monster are all at apartment 109.

In each of these exercises, the apartment starts with all six puppets attending the party. Each of the puppets enters and leaves the apartment for particular reasons, and sometimes new puppets enter. After each numbered exercise, write down the contents of the apartment; in other words, write down who is left in the apartment after all the incidents.

1. Ernie accidently spills milk, but Elmo doesn't have a mop available. Bert runs out to buy a mop.
2. Cookie Monster eats all the cookies, and goes to Mr. Hooper's store to borrow some cookies. The Count decides to count the number of balloons Elmo has hanging up, and so he joins the party. Telly leaves to invite Baby Bear.
3. Elmo leaves to get some squirt guns, and Bert responds to a call to retrieve a kitten from a tree. Ernie leaves to go to his apartment and grab some water for the kitten. By the time Ernie hands the kitten the bowl of water, Elmo has returned to the party.
4. Snuffy tries to enter the doorway and gets stuck in the doorway, though he is considered "inside the room." Telly leaves out the window to get some help, and Cookie Monster joins him. Oscar climbs in the window bringing some grouchy treats to the party. Bert can't stand the stench and jumps, not climbs, out the window. Snuffy is able to get out of the door before anyone else returns, and he decides to leave the party.

ANSWERS:

1. Elmo, Telly, Ernie, Cookie Monster and Grover are in Elmo's apartment.
2. Elmo, The Count, Ernie, Bert and Grover are in Elmo's apartment.
3. Elmo, Grover, Cookie Monster and Telly are in Elmo's apartment.
4. Oscar, Elmo, Grover and Ernie are in Elmo's apartment.

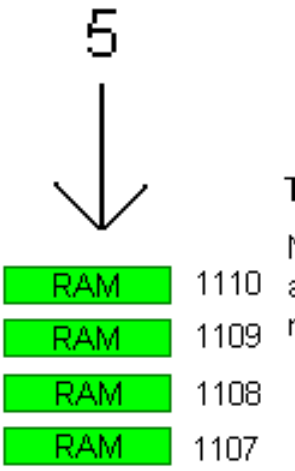
Now how does this apply to ASM programming? As you remember, the calculator's RAM is used for storage. The question is, how does the calculator know where everything is? You could put a picture in RAM, but the calculator needs to know where that picture is located. If you don't know where Elmo's apartment is, you can't find Elmo and you'll get lost. If the calculator doesn't know where in RAM your picture is stored, it won't find it.

Recall that you can find the puppets in this example by looking for their apartment number, their address. Every byte of RAM in the calculator has an address, a location, designated by a number just like the apartments of the puppets. So by going to a particular address, the calculator can access a certain part, a certain area, of RAM. By telling the calculator the address your picture is located at, the calculator can find the picture and act accordingly.

Let's apply our knowledge to an example. Suppose you want to design a game that has a number of lives. To keep this value permanent, you need to keep this in RAM. However, the calculator has to know where this value for the "Number of Lives" is. Let's pretend that you chose to keep this value for the number of lives in the 1110^{th} byte of RAM. In this case, you're using a **pointer;** that is, you're pointing to the 1110^{th} byte of RAM, where the number of lives is located, or where you WANT it to be located. (For example I can show you where Elmo is by "pointing" my finger at apartment 109) Later on, you will learn exactly how to tell the calculator where the number of lives is.

ONE:

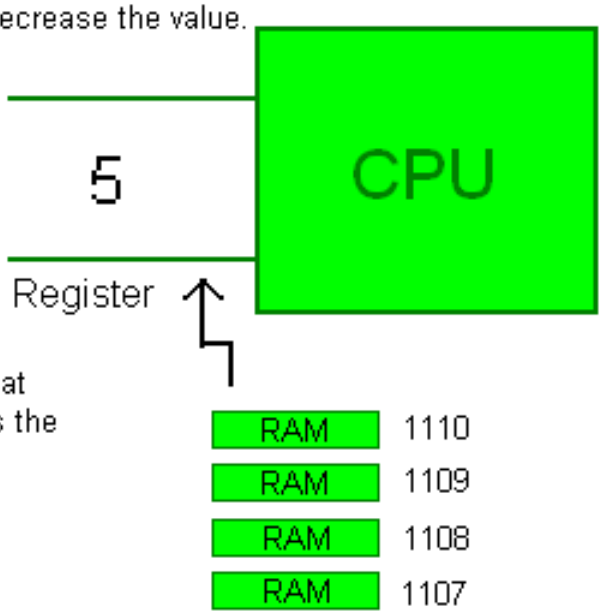
Let's assume that the player of your game starts with 5 lives. The calculator needs to remember that the player has 5 lives, and so it needs some place to store it. You tell the calculator to store the number of lives in address 1110.



TWO:

Now the RAM defined at address 1110 contains the number 5.

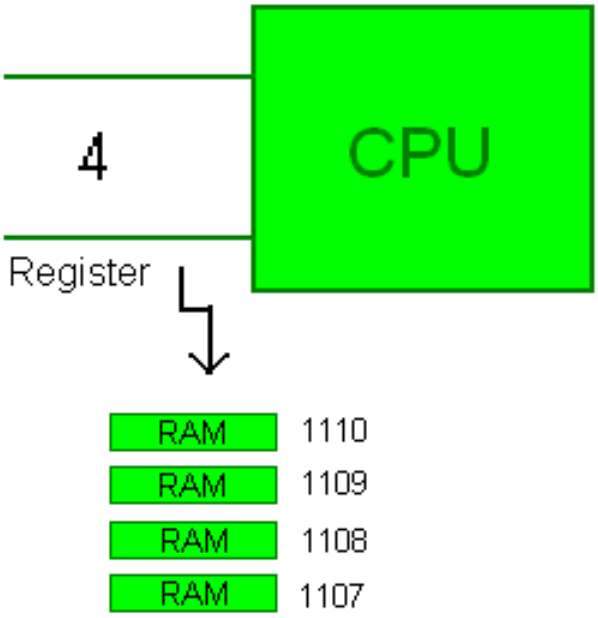
THREE: Suppose now that you need to decrease the number of lives to 4 lives. Remember that RAM is used for permanent storage, not for operations. So you tell the calculator to retrive the number of lives, which is at address 1110, and store it in a register so that the CPU can operate on it and decrease the value.



RAM = One Byte of RAM

FOUR:

Now that you have the number of lives, you can store it back in RAM so that the calculator can remember it in the future. Since you're using the 1110th byte to store the number of lives, you store the number of lives in address 1110.



MORE ABOUT RAM

Although the Ti-83+ has 24 KB of RAM that you can use for anything, there is other RAM available that the calculator uses. How does the calculator remember the text you entered on the home screen when you enter a menu? How does the calculator know where the cursor is located? How does the calculator remember what the Xmin, Xmax, Ymin and Ymax values you entered are, even though you don't find these variables in the 24 KB of user RAM? The answer is, the calculator has some RAM, called **System RAM,** reserved for these variables, for the text screen, and other states required. And like all RAM, each byte of System RAM has an address. As you program in ASM, you'll learn to access this RAM to draw/save pictures, change the viewing window, and place the cursor in a random position on the text screen.

LOOKING AHEAD

If you've made it this far, congratulations! You will be ready to start programming in ASM next tutorial. From this point on, there will be more exercises and more examples per tutorial to help keep you on the right track. In other words, I won't let you lose your footing.

In your spare time, check out this game:
http://www.kongregate.com/games/Moly/gorillas-bas

While you will be writing many small programs to practice what you've learned, you'll also be, little by little, porting this game to the calculator, kind of like a culminating project…it will feature menus, text, full-color graphics, and even sound! Don't worry, I'll be guiding you the entire way to write this game.

Ready to begin? You'll write your first Z80 program in tutorial number 4.