The Final Milestone: Tracking a white object on a black background

An image sensor is a device that converts light into electrons. One simplified way to think about the sensor used in a digital camera is to think of it as having a 2-D array of thousands or millions of tiny solar cells, each of which transforms the light from one small portion of the image into electrons. When the camera's shutter opens, light is directed through the lens to the sensor and impinges on the tiny solar cells. The more light that strikes the tiny solar cell, or *pixel*, the higher the number of electrons produced. There are several transistors at each pixel that convert these electrons into an analog voltage. An analog-to-digital converter turns each pixel's voltage value into a digital value proportional to the amount of light that struck the pixel. There is circuitry on the sensor that reads the digital values for each pixel, starting in the upper left corner, column by column and row by row until the bottom right end of the sensor is reached. The digital values for the sensor used in our project are 8 bits each. These values are made available on the pins Y7 through Y0 on the camera module. A digital value of 0xF0 corresponds to a white pixel and a value of 0x10 corresponds to a white pixel with shades of gray in between.

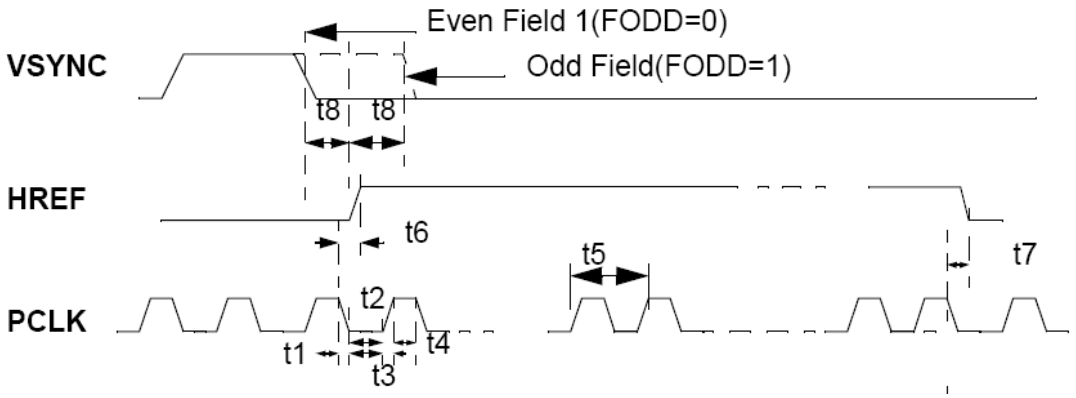
The entire 2-D array of pixels is referred to as a frame. Each time a new frame begins, the image sensor produces a pulse on a signal called VSYNC (Vertical sync). The 0 to 1 to 0 transition indicates that the sensor is about to send out a new frame beginning with the pixel in the upper left corner of the sensor (Refer to the timing diagrams below). The frame is composed of 288 rows of pixels, with each row consisting of 352 columns. Each time a new row begins, an active HIGH level is produced by the image sensor on a signal called HREF (Horizontal reference). All the pixels of a given row are sent out on Y(7..0) while HREF is HIGH. Each time a new pixel is sent out, the image sensor produces a pulse on a signal called PCLK (Pixel clock). The pixel is made available on the rising edge of PCLK and *should be sampled on the falling edge of PCLK*. Refer to the wiring diagram provided previously. This diagram indicates which I/O pins are used to carry the above mentioned signals. Also note that interrupts can be generated on each of these I/O pins. It will be left to you to decide whether to use interrupts or polling on these timing signals.

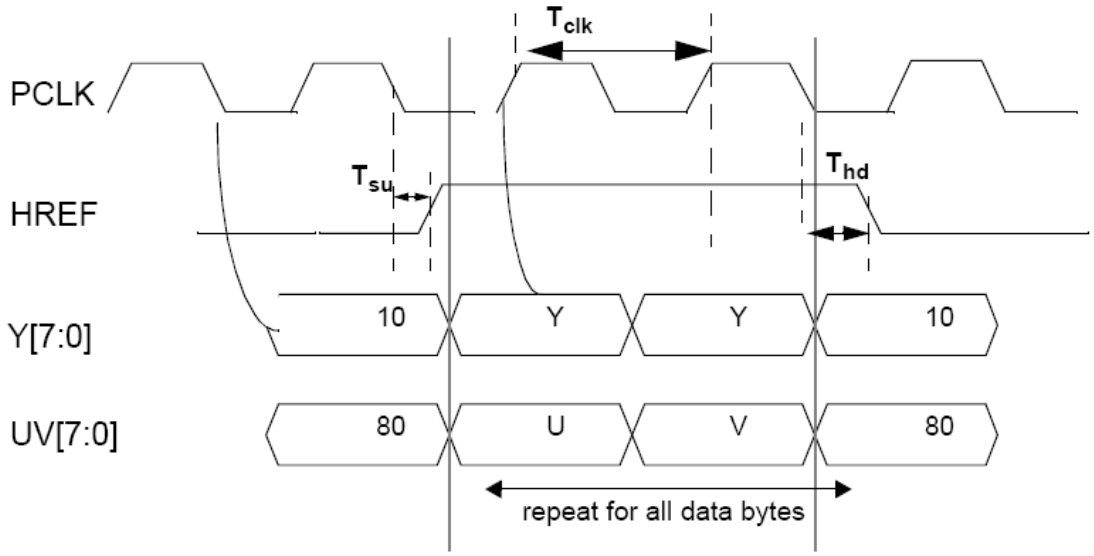
In this project, resolution is not all that important. A command can be sent via the TWI to tell the image sensor to reduce the resolution from 288x352 to 144x172. Since the time per frame is constant, this effectively increases the time between pixels. The time between pixels can be further increased by sending a command to the image sensor telling it to reduce the pixel rate. Another command can be sent to the camera to tell it to generate the PCLK only while HREF is HIGH (The default is that it constantly generates PCLK). Because the tracking function requires some processing to be done between pixels, you probably will have to take these steps (You must calculate the time between pixels to determine how much time you have for processing). You will have to examine the image sensor data sheet to determine which registers to send the commands to as well as the value that should be sent.

Tracking is by far the most difficult of the milestones so far. The strategy you use is entirely up to you to determine. The strategy revolves around keeping the camera pointed at the center of the

white square. As the square is moved, the turret should move to allow the camera to locate the center of the square. This will probably require you to add new functionality to your servo API since you must make movements that are relative to the current position. You must calculate the center coordinates of the square and compare them to the center of the camera's field of view (ideally 72, 88). The difference between the two is the error signal. This error signal is then used to command the turret to pan and/or tilt to the correct location. You are attempting to minimize the error signal. After a movement is made, the new center of the square is calculated and a new error is also calculated. This is again used to make adjustments to the turret position. This is repeated until the error is reduced to (almost) zero.

The final project must be capable of implementing all of the commands required in the previous milestones. In addition, you will now also include a command called TRACK. Tracking is done in an infinite loop, so, once tracking has begun, a reset of the board will be required to stop it.





Pixel Data 16-bit Timing
(PCLK rising edge latches data bus)