| **VIDEO P6-2-5b-CapSense Version 2** | | |
| --- | --- | --- |
| Cell | **Visuals** | **Audio** |
| 1 |  | Welcome back to Cypress Academy, PSoC 6 101. In this video, I will show you how to take the CapSense implementation we learned about in the previous lesson and add it to our BLE-controlled robotic arm project. |
| 2 | SCREEN CAPTURE:  P6-2-5b-CapSense\_capture1.trec | Let’s start by copying the CapSense component from the Basic project we just did and pasting it into the MainController schematic.  Next, verify the pin settings by clicking on the pins file in the design wide resources. All the pins should be the same as the Basic project. Then we will generate application. |
| 3 | SCREEN CAPTURE:  P6-2-5b-CapSense\_capture2.trec  (capsenseTask.h) | For the firmware, we first create capsenseTask.h with pragma once and the function prototype for the capsense task. |
| 4 | SCREEN CAPTURE:  P6-2-5b-CapSense\_capture3.trec  (capsenseTask.c) | Next create capsenseTask.c. You will need includes for project.h, pwmTask.h and global .h.  It will only have the capsenseTask function which is defined the same way as all of our other tasks.  The slider is going to control the motor position of a given motor. I’ll use the buttons to select which motor the slider is changing.  So, create a variable called currentMotor to keep track of which motor we’re changing. Create a PWM message. This is one of the cool things about an RTOS, you can have multiple, independent tasks sending messages to other tasks—in this case CapSense and UART are both sending PWM task messages.  Now, same as before, we’ll start the CapSense component and scan the widgets.  In the infinite loop, when the CapSense hardware is not busy, we’ll process the widgets.  Next, we’ll find the position of the linear slider. If the user is touching the slider, we’ll build a message that contains the slider position and which motor we’re changing; and send it to the PWMQueue.  Then depending on which button is being touched we’ll change the currentMotor variable to change which motor we’re tracking.  Now, update the baselines and start the scanning again. |
| 5 | SCREEN CAPTURE:  P6-2-5b-CapSense\_capture4.trec  (main\_cm4.c) | Lastly, you need to start the CapSense task back in main\_cm4.  Build, Program and test. |
| 6 | VIDEO:  Show video of ARH pressing buttons and using slider. Need to be able to see robot arm moving in background or as an inset picture or split screen. | Now as I press button0, I can run my finger on the slider and change the position of motor 1; if I press button1, I can change the position of motor 2. Excellent! |
| 7a | SCREEN CAPTURE:  P6-2-5b-CapSense\_capture5.trec  (show Tracealyzer window) | **Option 1:**  Now we have CapSense working to control the robotic arm on the PSoC 6 BLE Pioneer Kit. In the next video, I will walk you through an extremely useful tool to debug RTOS applications using a tool called Tracealyzer by Percepio, a partner of ours. |
| 7b | VIDEO:  Show video of Alan waving the remote control around and the robot arm moving in the background. | **Option 2:**  Now we have CapSense working to control the robotic arm on the PSoC 6 BLE Pioneer Kit. In the next set of videos, I will walk you through how to enable the BLE connectivity to begin controlling the robotic arm remotely. |
| 8 | TEXT ON SCREEN:  Cypress Developer Community  community.cypress.com  VIDEO:  Show video of ARH email and twitter windows. | You can post your comments and questions in our PSoC 6 community or as always you are welcome to email me at alan\_hawse@cypress.com or tweet me at @askioexpert with your comments, suggestions, criticisms and questions. |