SOFTWARE GUIDE: CVC

CVC: E128 Software	
AD_E128.c **AD_E128.h**	Provides analog-to-digial reading ability. used to convert the analog potentiometer signal to a digital one.
Init.c **Init.h**	Provides hardware-specific initialization functions for pin directions, communication set up, etc.
PacketBuilder.c **PacketBuilder.h**	Protocol outgoing layer, takes a given message and assembles a byte sequence the XBee can transmit.
PacketInterpreter.c **PacketInterpreter.h**	Protocol incoming layer, takes a given sequence of received bytes and extracts the relevant message.
SMEvents.h	State machine event definitions for each of the state machines
SMGame.c **SMGame.h**	State machine for the overall game play. the highest-level module
SMReceive.c **SMReceive.h**	State machine for receiving wireless data
SMTransmit.c **SMTransmit.h**	State machine for transmitting wireless data
UltrasonicSensor.c **UltrasonicSensor.h**	Provides functions for initializing the ultrasonic sensors and requesting and retreiving data. designed with a pseudo state machine.
Timer_E128.c **Timer_E128.h**	Provides up to 8 timers using the real-time-interrupt (RTI) register of the E128.

SOFTWARE GUIDE: ACV

ACV: Xbee PIC16 Software	
Pic16Zigbee.c **Pic16Zigbee.h**	This c-file contains the main() function for the AVC Zigbee PIC. This c-file also contains all setup and initialization functions as well as the check events function. The h-file is the master header file of this PIC and links to all other header files.
SMEvents.h	This h-file holds all the event types for all state machines on this PIC.
218ZigbeeProtocol.h	This h-file holds constants for bit positions and information in the ME218C protocol created by our class.
SMRobustComm.c **SMRobustComm.h**	This state machine implements timers and checks events pertaining to finding teammate and capturing an atoll to insure that multiple transmissions are carried out when appropriate.
SMSpiIO.c **SMSpiIO.h**	This state machine handles SPI reception from the RFID Master PIC and raises flags for the appropriate events.
SMReceive.c **SMReceive.h**	This state machine handles UART reception from the Zigbee and raises flags for the appropriate events.
InterpretPacket.c **InterpretPacket.h**	This module is called by the SMReceive state machine and holds functions to parse/interpret received UART data and store it so it can be read later by other modules.
SMTransmit.c **SMTransmit.h**	This state machine transmits UART to the Zigbee for wireless broadcasting.
BuildPacket.c **BuildPacket.h**	This module is called by the SMTransmit state machine and holds functions that assemble transmission messages according to the ME218C protocol created by our class.
ACV: RFID Assembly PIC16 Software	
RFID_PIC.asm	This program receives data from the RFID receiver, decodes it into HEX serials, and sends it to the master PIC via SPI.
ACV: RFID Master PIC16 Software	
RFIDSM.c	This is the code for the RFID state machine. It reads RFID serials from the RFID PIC, and determines whether they are atolls or team cards.
SecConSM.c **SecConSM.h**	This is the state machine that runs from the MASTER_PIC and interfaces with the security controller. It sends RFID bytes to the SC and receives the encrypted security keys back.
Initialization.c **Initialization.h**	Initializes all of the master PIC hardware, including UART and SPI
EventChecker.c **EventChecker.h**	This is the event checker for the MASTER_PIC. It has a public function, CheckEvents(), that checks various inputs for communication events and throws out events that trickle down to the state machines.
XBeeSM.c **XBeeSM.h**	This is the state machine that sends messages to the XBee Pic.
SPI_SM.c **SPI_SM.h**	This state machine sends and receives messages over SPI. It must be run on the master.
ACV: Motor PIC16 Software	
EventChecker.c **EventChecker.h**	This module contains the CheckEvents function, which returns receive events and motor enable/disable events.
InterpretPacket.c **InterpretPacket.h**	This module interprets incoming packets, and if they are motor speed packets, extracts the relevant PWM data from them.
MOTOR_PIC.c	This is the main program for the motor PIC. It runs the Receive State Machine, which receives XBee messages over the UART and determines whether they are for the motor PIC. If they are, the PWM values are extracted and written to the PWM register.
Motors.c **Motors.h**	This is a low level module that controls the initialization and operation of the PWM and UART systems on the PIC.
SNReceive.c **SNReceive.h**	This module implements a state machine to receive asynchronous UART communication using the ZigBee API Frame with ME218C 2011 Communication Committee's official protocol (see documentation in ME218C 2011 Final Project)