Indice

Capitolo 1

1.1 Phase 1

1.1.1 Hardware

- Scorbot robotic arm.
  This five degrees robotic arm consists in five 12V-(0.5-1A) motors plus the motor driving the gripper. They move respectively base, shoulder, elbow joints and the last two motors are coupled handling the pitch and roll movements. A quadrature optical encoder has been mounted on the axis of each motor in order to measure the angular position. There is also an end switch sensor for each motor due to obtain the initial position (usually called home position) for the robotic arm in which encoders can be zeroed. The interface is a D50 pin connector containing direct wires with motors, encoders and end switches.

- STM32F4-discovery

  - TIM6: drives the HighResolutionTimer and can be used as a synchronisation point for the execution cycle.
  - TIM4, TIM9: drives the PWM generators. Four channels for TIM4 and two channels for TIM9 can handle all the six motors of the scorbot.
  - TIM1: can be used as clock source for external encoder counters. The pwm generator channel 4 has been enabled.
  - TIM(2,3,5): have been used as internal encoder 16-bit counters with no interrupts.
  - ADC1: five channels 2-3-4-8-9 have been activated in order to measure the current dissipated by the five motors for a further torque control.
  - SPI1-SPI2: have been enabled in order to manage further external communications with serial encoder counters.

  - D0-D3 pins: set as output pins have been used for OE and SEC pins for the two external encoder counters (HCTL2022 chip).
  - E7, E8 pins: RST active low reset pins for the two external encoder counters (HCTL2022 chip).
  - B(10,11)-D(8,9,10,11) pins: set as output pins they have been used to set the direction of the six motors.
  - E9-E13 pins: set as input pull-up pins they have been set to read the microswitches inputs (low active).
  - C0-C15 pins: set as input pull-up pins they have been set to read two 8-bit ports from external encoder counters (HCTL2022 chip).
  - USART2: pins D5=Tx and D6=Rx is used as the error stream flushing error messages.
  - USB-OTG-FS: uses the pins A11 and A12. This stream is used to get the RT diagnostic from any device listening on the USB port and to get eventual data such as references or control signals directly (depending by the application).

- HCTL2022: is an external parallel 32-bit encoder counter. The OE pin is active low disabling the writing on the chip internal registers in order to read the counter in a stable mode and there are two SEL pins used to select the four bytes to be read. Now a single SEL pin has been used because a 16 bit counter is sufficient for this application. The other SEL pin has to be grounded. As we can see in the STM32 pinout a pwm generator has been set in order to provide an external clock to this chip. Three internal encoder counters have been activated in the STM32 board, so we need two more external counter in order to handle the five scorbot motors without using interrupts disturbing the real time execution.

- L298N H-bridge: each one of these can be used to drive two DC motors with less than 2A continuous current. For the scorbot three of them have been used to handle the six motors. For each motor we use an output pin selecting the motor direction and a pwm signal for the motor speed. The current strategy to drive the motor is low direction pin and normal pwm for left-to-right direction and high direction pin and the opposite of pwm signal for right-to-left direction. This strategy minimizes the number of pins used to drive each motor.

1.1.2 Software

The first developed application test aims to control a single motor of the scorbot. Thus the RobotArmGAM has been developed in order to implement the control system. The first signal is the timer (TIM6) for synchronisation with a frequency of 50

Hz. Follows the signal reading from USB in non-blocking mode the reference of the motor. Currently it receives a couple of values: the number of the motor to be controlled and the desired position in encoder steps. The third signal reads the encoder counter from the internal STM32 timer. The last input signal reads the micro switch value from an input pin.
The output signals are the motor direction pin, the pwm control signal and the diagnostics signal to be sent out to a device listening on the usb port.
The input signal used to receive the motor references can be used also to change the GAM state. In particular if the first value is -1, the second value selects the next state to be executed.

- 0 - basic application: if the second value is a positive number 12V will be applied to the motor, if negative -12V. Basically this state has been designed in order to move manually the robot.
- 1 - home positioning. -12V will be applied to each motor until the end switch sensor becomes active because the motor has reached a fixed position.
- 2 - control application. The second value specifies the desired encoder position of the motor. The parameters of a PID controller can be configured from the configuration file.
- -1 - turn off the application stopping the scheduler execution. The application can be restored changing the state to one of the previous ones.

The current application sends on usb port in output reference, encoder, pwm, cycle time, ecc. On the other side a pc can be used with MARTe where the InputGAM USBDrv has been developed in order to provide a friendly user interface thanks to the HTTP interface. It is possible send the two values with desired joint position (motor number and encoder reference) to the board and each cycle the diagnostic values can be visualized in real time or stored also in matlab format.
GAMs:

- RobotArmGAM: Follows the configuration parameters
  - NumberOfMotors: the number of motors
  - SwitchPinMask: the mask of input pins for the MicroSwitch sensors. We assume all these input pins belonging to the same port.
  - Kp: an array with the proportional gain of each motor controller.
  - Ki: an array with the integral gain of each motor controller.
  - Kd: an array with the derivative gain of each motor controller.
  - MaxPwm: an array the max pwm period for each motor

  - MinPwm: an array the min pwm period for each motor
  - MaxControl: an array the max control for each motor (in mV) The control will be saturated to be in [-MaxControl, -MinControl]
  - MinControl: an array the min control for each motor (in mV). It will be used to determine the motor control deadzone.
  - EndSwitchBound: how many cycles to wait before setting control to zero because of a deadzone.
  - MotorDirection: can be used to invert the moving direction of the motors for positive or negative voltages.
  - CounterDirection: should be set accordingly to motor direction in order to increment or decrement the counter when moving on what is considered positive or negative direction.

  Signals:

  - TimerSignal (input): connected to TimerDataSource is the synchronization point currently set to 50Hz.
  - FromUSB (input): connected to USBCommunication reads in non-blocking mode the couple [motor number, motor reference] at each cycle.
  - FromEncoder1 (input): connected to EncodersCounter_opt reads the encoders from the three internal STM counters.
  - FromEncoder2 (input): connected to HCTL2022 reads the encoders of the fourth motor from the external HCTL2022 encoder counter chip.
  - FromEncoder3 (input): connected to HCTL2022 reads the encoders of the fifth motor from the external HCTL2022 encoder counter chip.
  - FromMS (input): connected to GPIO reads the value in input from Micro-Switch sensors.
  - ToPwm (output): connected to L298N sets the pwms value and the motors pin directions.
  - Diagnostics (output): connected to USBCommunication writes on USB interface reference, pwm and encoder counter for each motor.

Some new DataSources implemented just for this project should be described:

- EncodersCounter_opt: reads the internal encoder counters. Basically is a read from the TIM->CNT register, but has been implemented in vector mode in order to handle a group of internal counters. It must read the timer numbers from configuration.

- HCTL2022: handles the external counter HCTL2022 chip. Follows an example of the configuration which must be provided:

```
Class = HCTL2022
OE_Port = GPIOD
OE_Pin = 1
SEL_Port = GPIOD
SEL_Pin = 0
RESET_Port = GPIOE
RESET_Pin = 7
CLK_Id = TIM1
CLK_Channel = 3
ByteReg_Port = GPIOC
ByteReg_Mask = 0xff
```

  note that we must set the port and pins connected to the HCTL2022 pins such as OE (low when read high otherwise), SEL (to select the byte to be read), RESET (to reset the counter), CLK (external clock to the HCTL2022) and the 8-bit pin set to read the byte one by one.

- L298N: handles the motor driver. Follows an example of the configuration which must be provided:

```
"            Class = L298N"
"            TimIdentifiers = {4, 4, 4, 4, 9}"
"            Channels = {0, 1, 2, 3, 0}"
"            DirPorts = {1, 1, 3, 3, 3}"
"            DirPins = {10, 11, 8, 9, 10}"
"            MaxPwm = {999, 999, 999, 999, 999}"
```

  note that we must provide the timers providing the pwm signals and the direction pins for each motor. The max pwm period is necessary in order to calculate the inverse pwm wave for the negative direction.

An important note is that the encoder counters are all in 16 bit, but the used type is an int32. With a trick we can obtain a 32-bit counter (int32). Basically we store an int32 encoder value and at each cycle we sum the difference.