This package provides the low-level interface to the AVR. Higher-level abstractions used by application code will access this package, although application code may access this package directly when necessary (e.g., to call the Power_Down routine). All access to the current button and raw inputs are through this package.

Note that this package initializes the AVR automatically.

with NXT.AVR_IO; use NXT.AVR_IO;
with Interfaces; use Interfaces;

package NXT.AVR is
pragma Elaborate_Body;

procedure Power_Down;
-- Send a power-down message to the AVR

procedure Set_Power
  (Motor : Motor_Id;
   Power : PWM_Value;
   Brake : Boolean);
-- Send motor control message to the AVR

procedure Await_Data_Available;
-- Wait until at least one set of messages has been sent and received
-- from the AVR, such that the data below are now available to be accessed.
-- To be called once, prior to accessing the sampled values declared below.

-- The following are the sole means of acquiring the values. In other words, abstractions and application code should not interact with the AVR to get them. These objects are to be treated as strictly read-only.
-- They are updated periodically by an task internal to this package. Any other updates will be overwritten by that task. Access to the individual
-- values is thread-safe.

Raw.Buttons : Unsigned_16;
pragma Atomic (Raw.Buttons);
-- The most recent raw button readings received from the AVR.

Raw.Input : Raw_ADC_Inputs;
-- The most recent A/D input readings received from the AVR for the four
-- sensors. The type Raw_ADC_Inputs has Atomic_Components applied.

Raw.Battery : Unsigned_16;
pragma Atomic (Raw.Battery);
-- The most recent battery voltage reading received from the AVR. Units are
-- millivolts. The bit indicating use of rechargeable batteries is
-- included in the value.

procedure Set_Input_Power (Port : Sensor_Id; Power_Type : Sensor_Power);
-- Control the power supplied to an input sensor

end NXT.AVR;