Jacob Harris Embedded Systems

4/2/2015

Milestone Alpha: Economical Linear Actuator with Wireless Networking

Milestone Alpha entails software documentation and planning as well as proof of concept of key hardware components. All software module header files and doxygen documentation to be written by the team-member are attached with this document. Table 1 below shows an overview of the three modules, and the functions documented as of this milestone. The first module has been renamed from “pulses” to “stepper\_drive\_hal” since its purpose was to allow the stepper\_drive module to remain platform-independent, and the purpose of the “pulses” function was more or less to abstract the hardware layer. Next, all function names have been clarified with a prefix, making the same

Hardware has been prototyped, with a Pololu© stepper motor breakout for the A4988 micro-stepping driver and a 200 pulse/revolution NEMA 11 stepper motor. For the assembled linear actuator, a higher torque NEMA 14 motor may be desired, which should also be compatible with the above driver. Tentative specifications for the linear actuator remain the same as the proposal. Specifications of the prototype linear actuator are (subject to change due to availability of parts) as follows:

* Linear range of motion: at least one foot
* Load capacity: at least ½ pound at slow speed
* Linear speed (max): at least 4 inches/second
* Motor type: stepper motor, bipolar
* Rail system: 8 mm round rod with linear bearings
* Microcontroller: Ti MSP430 series
* Radio communication nRF24L01+ radio

Documentation for the following three modules (Table 1) has been completed: stepper\_drive, stepper\_drive, motion planner. All other modules for use in this project are provided. Modules planned to be used, but not personally documented by team are as: network, SPI, nRF24L01+, task, timer. Network, SPI, and nRF24L01+ are as of yet unimplemented. Testing can continue without them, as they only provide network connectivity, and no communication is needed for proof of concept in driving the motor or generating the desired velocity curves.

|  |  |  |
| --- | --- | --- |
| **Module name** | **Description** | **Functions** |
| **Stepper Driving** | | |
| stepper\_drive\_hal | Interfaces with timer peripheral to generate pulse trains of varying frequency and length | void timerInit();  void pulse\_start();  void pulse\_stop();  void pulse\_setPPS(uint16\_t PPS);  void pulse\_update(void(\*fn)void); |
| stepper\_drive | Calculates shape of stepper velocity curve, sets appropriate speeds using the pulses module, keeps track of position | void stepperInit (uint16\_t max\_SPS, uint16\_t min\_SPS, uint16\_t SPS2);  void stepper\_start();  void stepper\_stop();  void stepper\_setSPS(uint16\_t SPS);  void stepper\_moveTo(uint16\_t position);  void stepper\_update();  uint16\_t stepper\_numSteps(uint16\_t SPS\_start, SPS\_finish) |
| motion\_planner | Queues position commands. Can ignore points between two other points if desired. e.g. A🡪C not A🡪B🡪C | void motion\_planner\_Init(uint16\_t mode);  uint16\_t motion\_planner\_addToQueue(queue \*motion\_queue, uint16\_t position, uint16\_t time); |

**Data structure for network based command of linear actuator:**

**<command> <linear data> <time data>**

**<1 byte> <2 bytes> <2 bytes>**

1. **command** - 1 byte unsigned int
   1. 0x00 – home actuator (move to zero position) ASAP
   2. 0x01 – move to position given in **<linear data> ASAP**
   3. 0x02 – increase current position by **<linear data> steps ASAP**
   4. 0x03 – decrease current position by **<linear data> steps ASAP**
   5. 0x11 – move to position given in **<linear data> in <time data> ms**
   6. 0x12 – increase current position by **<linear data> in <time data> ms**
   7. 0x13 – decrease current position by **<linear data> in <time data> ms**
   8. **0xEE – reports current position in the following format: <2 bytes uint>**
   9. **0xFF – E-STOP**
2. **linear data** - 2 byte unsigned int
   1. linear data, in units of “steps” to be determined based on hardware
3. **time data** - 2 byte unsigned int
   1. time data, in units of milliseconds (ms)