square

Proposal and Milestones of a Hardware Prototype to sustain a Daisy Chain Protocol Development in Force-Guiding Particle Chains for Shape-Shifting Displays

Raoul Rubien, BSc
rubienr@sbox.tugraz.at

***Abstract*—What is a particle chain? A particle chain is a specific kind of programmable matter. Programmable matter are simple robots that are connected together and have to interact with each other. It usually consists of a high volume of small scaled particles that can be rearranged to achieve specific shapes. To communicate with particles of a force-guided particle chain LasagniR14, a state of the art one wire communication (Dallas One-Wire™[1]) is applied. Thus when the overall current consumption exceeds the Dallas One-Wire™maximum specification the communication bus can not operate faultless. That is why the development of a new protocol is required. To sustain this later work we elaborate the design and assambly of a particle pototype and also a tool chain that operate as a whole.**

I. Introduction

Unlike several in literature proposed folding methods Knaian2012,Hawkes2010,M the force guided chain does not utilize motors, magnetic adherence fields or origami folding to achieve folding. The force guided chain is made of tied particles that can be folded in two directions of one geometric dimension similarly to a snake's shape when moving on a plane surface. With parallel mounted chains surfaces or 3D models can be approximated.

How it works: All particles in our free-hanging particle chain are naturally pulled down by gravity. Thus the chain's particles are ordered in a line. Between neighboring particles two connections, one at the left and a second at the right side, are realized with monostable hinged edges. When the chain is straight, the edges reside in a locked state. In this state an utilization of the external force would lead to a contraction of the chain but will not shape it. Only if hinges are unlocked in advance the force would lead to shape forming. As an example let us see what happens if the chain is contracted but one left hinge in-between two particles is unlocked and all other are locked. The applied force contracts the chain and compresses the particle's left side (shorten the distance) whereas the right side remains at the same length. This ends in folding of two particles to the direction where the hinge was previously unlocked.

Hinges are unlocked by actuators made of Shape Memory Alloy[2] (SMA) wires that contract themselves when heated up. Assuming the contraction force and distance is enough hinges can be unlocked with that kind of wires. The SMA wires, having low resistance, can be exploited to realize data communication in-between them. Due to the underlying physical protocol, the current implementation makes communication infeasible when the total power consumption exceeds the 1-wire maximum rating. As a compromise of this fact transmissions and power consuming tasks can never take place simultaneously in a whole chain. They have to be processed sequentially.

The motivation of this work is to enhance the communication speed through decoupling communication from the power line. As a consequence the 1-wire will be replaced by a daisy-chain protocol. This overcomes the dependency on maximum current limit of the Dallas One-Wire™ bus.

Nevertheless while two adjacent particles are actuating their SMA wires, data cannot be exchanged. This slowdown is expected to be low in relation to the 1-wire time-division multiplexing (TDM) bus, since the daisy-chain protocol brings some advantages with it. Let us denote possible interactions at a given time slice between adjacent nodes $\langle P_m, P_{(m+1)}\rangle$ as $\{I_m^{tx}, I_m^{rx}, I_m^{actuating}\}$ then:

- $(I_m^{tx} \vee I_m^{rx})$ excludes $I_m^{actuating}$ and vice versa (during actuation no transmission and v.v.)
- $(I_m^{tx}$ and I_m^{rx} can be performed simultaneously
- $(I_m^{tx} \vee I_m^{rx})$ and $(I_{m+n}^{tx} \vee I_{m+n}^{rx})$ can be performed simultaneously (independent communication between nodes)

As an illustration of simultaneous interactions let the last two particles of a chain be $\{P_{u-1}, P_u\}$. Both can be prepared to unlock one hinge. During this action communication between tuple (P_{u-2}, P_{u-1}) is still feasible but also between (P_{u-3}, P_{u-2}) until (P_0, P_1).

II. Goals

As the new protocol's physical layer differs from the current one there is no chance to re-use the circuity. This circumstance forces us to build a new prototype that is able to sustain the upcoming work. The proposal's outcome will be a combination of hardware and software infrastructure that provides support for protocol development: i) Hardware related: A fully functional PCB project (schema and routed PCB) that can be

[1] https://en.wikipedia.org/wiki/1-Wire
[2] https://en.wikipedia.org/wiki/Shape-memory_alloy

chained. ii) Software related: A Unix based tool-chain using CMake; a test software that can be used to check freshly assembled boards for errors; a simple debug possibility and a convenient method to invoke test cases.

III. Requirements

With respect to the project constraints listed in table I the requirements can be listed as:

1) New prototype particles that can replace the old design and that are able to control the actuators (via MOSFET bridge) and provide the basic support for serial full-duplex communication.
2) Test particles have to offer the possibility of fast access to test points: rx, tx before and after the H-bridges and one extra spare test point.
3) Particles have to signal their state by means of LEDs: One LED signals the device status; other LEDs are connected to actuator wires to indicate activity on them.
4) Easy accessible programming interface.
5) A reset switch needs to be on board.
6) Actuators should be replace-able by 5V light bulbs with electrical characteristics similar to the SMA wires.

Project Constraints
low price: use cheapest MCU possible
use as less components on particle PCB as possible
new communication with no extra wires: exploit actuators for communication
synchronous actuator operation: multiple particles have to execute commands synchronous at given time
parallel communication and latch operation: power consumption must not affect the communication; communication between nodes may take place while other node's actuators are operating
automatic particle enumeration: after a chain is powered the protocol enumerates nodes automatically
routing: particle passes packages over if not dedicated to itself

Table I
GIVEN CONSTRAINS FOR THE WHOLE PROJECT

IV. Project Milestones

Upcoming work will roughly contain: The development of a new particle-prototype; a first-time prototype activation and a firmware build system. Concrete milestones of the project are development of a PCB in section V-A, a first-time prototype activation in section V-B, the programming of the MCU in section V-C, creation of a development tool chain in section V-D on the next page and for quality assurance out of the tool chain launch-able unit-tests in section V-E on the following page.

A. Printed circuit board development

To develop the new communication protocol which uses actuators as communication lines a custom PCB that is capable of switching the actuator working mode (communication or actuating) is necessary. Also effortless access to hardware is desirable to be less time consuming when analyzing the physical communication. A low price MCU needs to be chosen to fulfill the constraints. For that reasons a new prototype PCB needs to be developed and assembled. It should permit the developer to have fast access to several important test points and provide some visual signals as well. The work will include:

- The development of a particle PCB test unit using Eagle[3] that provides test-points and LEDs for indicating tx/rx/power and status.
- Multiple particle prototypes have to be chained with respect to tx/rx and power wires.
- Obtaining or drawing missing Eagle libraries of newly used parts such as MCU, H-bridge, connector, ...
- Preparation of an electronic parts list and finding a supplier for them.

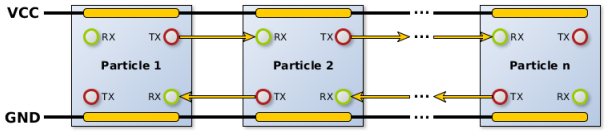

Figure 1. Chain of connected particles: Power supply can be soldered on a stable rail. Actuators may be replaced by 5V light bulbs.

B. Assembly and particle activation

Based on experience the first activation of a PCB/MCU can be a very time consuming task although it is not that complicated. It will contain following steps:

- Assembly
  - Accurate set up of the soldering-oven to achieve a correct temperature curve (temperature measure, visualization of captured data and comparison to manufacturer's recommendation)
  - The soldering of prototype units.
  - Fast quality check (look for shortcut, verify power consumption, verify possible misplaced parts) before activating particles.
- Particle activation
  - Writing and flashing a test firmware that drives all relevant IO-pins.
  - Verification of output on pins.

C. Programming the MCU

Per constraint definition in table I the usage of a low level MCU (i.e. ATTiny20[4] or lower) is desired. Since ATTiny20 provides rather low level functionality than other Atmega MCUs the focus will be on which tools can be involved for developing software for the MCU. This step will deal with:

- programmer
- programming interface protocol (TPI[5] or ISP[6])
- programming software (avrdude[7])

[3]http://www.cadsoftusa.com/
[4]http://www.atmel.com/devices/attiny20.aspx
[5]http://www.atmel.com/Images/doc8373.pdf
[6]http://www.atmel.com/images/doc0943.pdf
[7]http://www.nongnu.org/avrdude/

D. Setting up the tool chain

Software development is not just coding and executing also known as trial and error. It consumes much resources to achieve reliable and stable software that performs well. To avoid as many issues as possible a good assisting tool chain can be helpful. The time consumed at putting the tool chain together will later pay off by faster but also more qualitative development. The tool chain should at least be capable of:

- configuring the MCU, the programmer and the compiler settings
- compiling (i.e. *make*)
- storing the program to micro-controller and verifying: *make && make flash && make verify*
- pull statistics out of the source or hex (i.e. max heap size, control flow graph)
- running the program on simulator for debugging purpose
- starting emulated debug session with *make ddd*

E. Launching test cases

The ATTiny20 provides only a low level programming interface that is not capable of debugging. To avoid waste of time when searching errors and to increase code quality, unit-testing will drive the later development of the daisy-chain protocol. Therefore the tool chain must also be capable of running tests with less effort. Main points of that milestone will focus on:

- finding a convenient unit-test framework that can be used (if possible in combination with an AVR simulator)
- incorporate the framework in the tool chain
- running tests automated (i.e. *make tests*)

V. Future Work

A rough overview about available states of the daisy-chain-protocol is shown in figure 2. Beside the protocol development there is the question how a boot loader can be applied to allow the spread of a new firmware over a particle chain. Self programming particles would decrease the software deployment duration drastically.

The upcoming protocol itself is thought to provide a way to communicate to each chain's particle. Autonomously it does only enumerate the chain's participants. For applying the full protocol capability in chains a simple controller has to be put in charge. Such a controller, see figure 3, could be a simple Arduino[8] board. A controller does not only decouple timing sensitive communication from a chain, it also can be used as middle part for sending or receiving data from or for a computer or can be set up to run autonomously demonstrative chain procedures.

[8]http://www.arduino.cc/en/Main/ArduinoBoardLeonardo

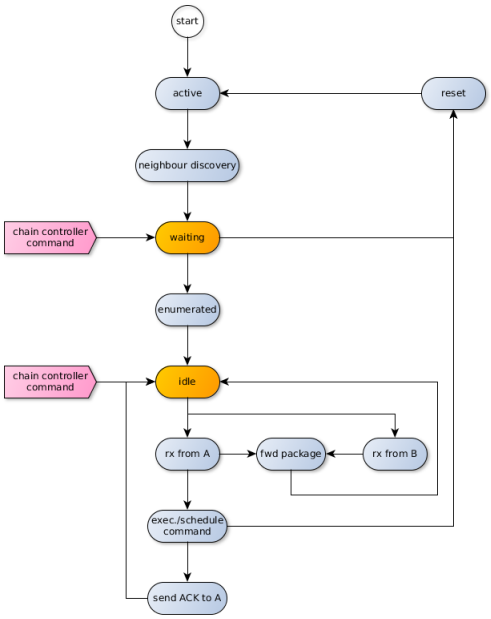

Figure 2. Proposed rough states of the upcoming protocol.

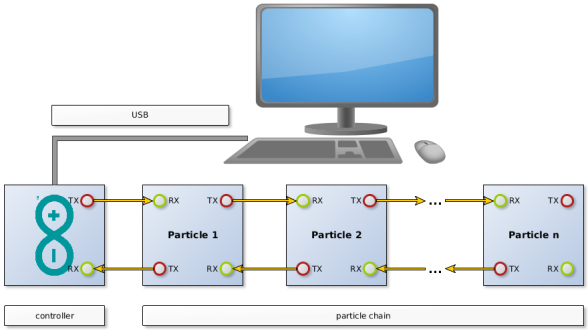

Figure 3. Illustrative example: Controller guided chain to decouple time sensitive communication. Controller applies the full protocol capability.

cite