CLOCKWORK

Programming with rules

Mike O'Connor

LATPROC

Language And Tools for Process Control

- First published 2012
- Linux hosted
- Open Source under BSD and GPL
- Developed a language which we call Clockwork

CAPABILITIES

Industrial IO (real world connections)

- BeckHoff EtherCAT(tm) based control

Humid (Human Machine Interface)

Protocols

- Modbus master/slave
- Raw TCP/IP and serial UART (device_connector)
- Plugins
- - libcurl
- - exec system
- MQTT

GOING BACK

GOING BACK

Wooltech - 40 years of automation

Electrical time and relay logic

Scorpion board John Langford and lots of extra boards

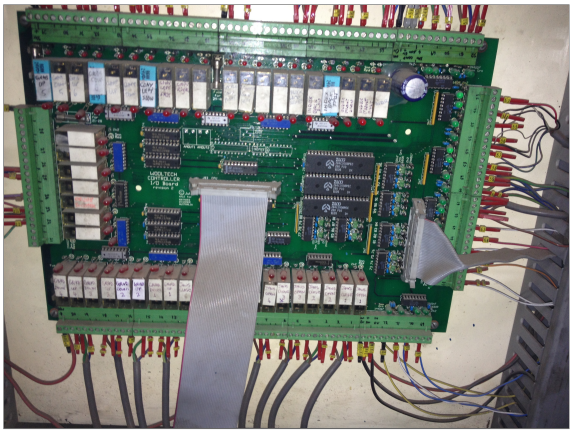
Custom Zilog Z80 board with inputs and relays

Early PLCs by Koyo (rebadged by GE and others)

Custom software on Linux (late 90s)

Back to Koyo PLCs during the 2000s using newer models

ISSUES WE EXPERIENCED



ISSUES WE EXPERIENCED

Long term custom hardware support is hard

Commerical solutions use binary file formats for source config

Industrial hardware is upgraded with forward only source formats.

External expertise often required

- Initially in building controller boards
- Later in writing control software

Hard to find industrially experienced coders

WHAT DID WE DO?

(the first attempt)

- Moved to a Linux based solution
- Programmed a solution with custom messaging and communications
- Programming took too long
- Result was not really what we wanted
- Program was not very reusable
- We discovered state machines
- We went back to PLCs

OUR REQUIREMENTS

Retain control

Reduce risk

Be future-proof

Reuse Don't Rewrite

Version control all source code

STARTING AGAIN

Simple programming language

Program pieces are called MACHINEs

Machines can be simulated easily

A Machines state resemble physical states and behaviour

```
LightSwitch MACHINE switch, light {
        on WHEN switch IS on;
        off DEFAULT;

        ENTER on  { SET light TO on; }
        ENTER off { SET light TO off; }
}
```

Controls 'light' based on the state of 'switch'

REDUCING RISK

By programming ourselves

By simulating as we develop

By reusing what we have done before

FUTURE-PROOFING

Use Linux

Use Open Source Software

VERSION CONTROL EVERYTHING

Proprietary, binary data formats make version control impossible

Over time PLC software or HMI panel firmware changes

Proprietary, windows-based programming tools do not maintain support for all versions

DEBUGGING

Effective debugging tools

- State Description
- Predicate explanation
- continues sampling of the state and properties changes.

CLOCKWORK

The latproc language

Objects are called 'Machine's

Describe Machines by their states

States are selected by evaluating rules

All Machines run continuously in parallel

Machines monitor each other

TOOLS

iod - main EtherCAT control daemon

cw - local interpreter daemon

iosh - terminal shell

sampler - monitoring and logging

persistd - monitor and retain persistend state

modbusd - bridge to modbus masters (panels)

device connector - bridge to TCP/IP or serial devices

TOOLS

iosh

```
> DESCRIBE O24V_GrabControlFan ;
---------------
O24V_GrabControlFan: off  Class: POINT
 instantiated at: /opt/latproc/code/config/config.lpc line:94
  parameter 1 module (EL2828_01), state: OP
  parameter 2 1 ()

 io: readtime: 185932 [Channel 2 Output, 1 0:1.1]=0 (0)
Exports:
  O24V_GrabControlFan
  published (1)
Listening to:
  EL2828_01[123]:   OP

Dependant machines:
  L_Inputs[399]:   nonempty
  M_ControlFan[580]:   off

properties:
  NAME:O24V_GrabControlFan, STATE:off, export:0, tab:Outputs, type:Output, wire:Y52

Timer: 20713486

>
```

TOOLS

sampler

```
443391 M_CO2 idle
443891 M_CO2 update
443892 XA_CO2 CO2_Level 496
443892 XA_CO2 message 496
443892 M_CO2 idle
```

USER INTERFACE TOOLS

Web UI - basic web applicationfor monitoring and control

humid - GUI development tool

- Buttons (toggle, momentary)
- Text and Number Entry Fields
- Page switching
- Element visablity control
- Time series graphing

web 3D visualisation - render a model in 3D

scope - character based graphing for sampler

OPEN SOURCE COMPONENTS

Linux, GNU compiler suite

cJSON, libmodbus, ZeroMQ

EtherLAB, libXML, three.js

nanogui, glfw

etc.,