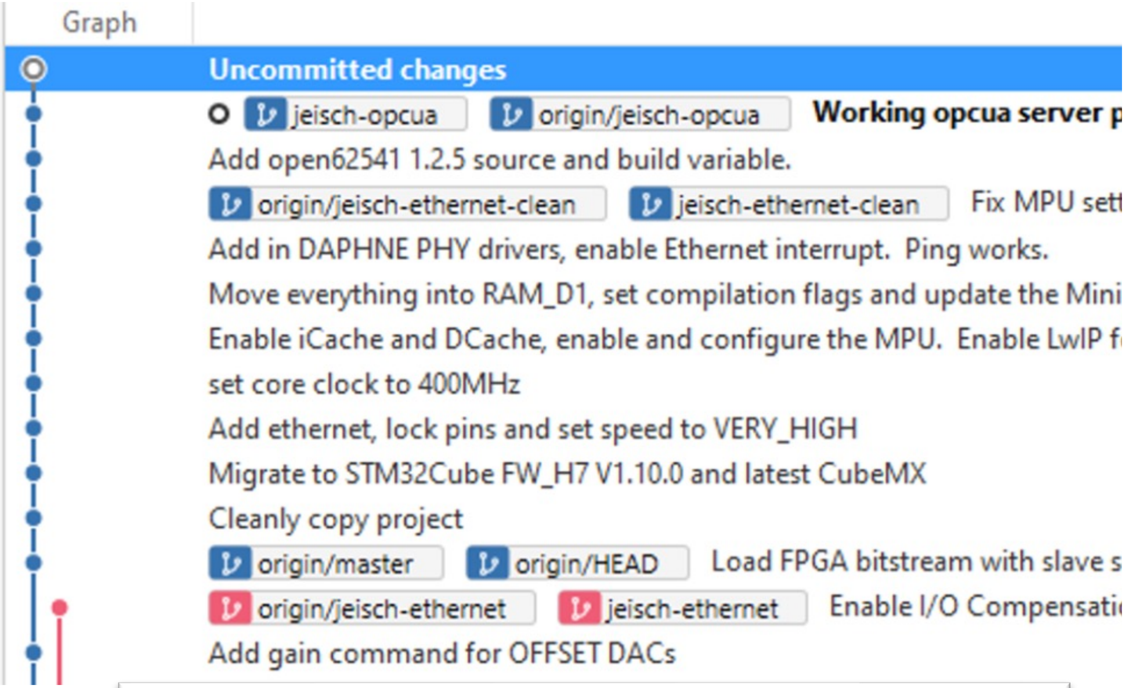
Status of Ethernet and OPC UA stack for DAPHNE

Jonathan Eisch, Fermilab

June 30, 2022

Development status

- **New branch starting with the latest master, including FPGA bitstream.**
- **Upgraded to CubeMX 6.5, FW_h7 V1.10.0 (including updated ethernet drivers) and CubeIDE 1.10.0.**
    - CubeMX 6.6 fixes various bugs, especially where it erronously enables DMA prior to enabling peripherals, but the code generation deletes a bunch of files (both generated and ours)
- **Ping works reliably to the static IP address.**
    - I need a command to read/write a MAC address &IP address/subnet mask from flash.



Graph

Uncommitted changes

jeisch-opcua | origin/jeisch-opcua | Working opcua server p

Add open62541 1.2.5 source and build variable.

origin/jeisch-ethernet-clean | jeisch-ethernet-clean | Fix MPU sett

Add in DAPHNE PHY drivers, enable Ethernet interrupt. Ping works.

Move everything into RAM_D1, set compilation flags and update the Mini

Enable iCache and DCache, enable and configure the MPU. Enable LwIP f

set core clock to 400MHz

Add ethernet, lock pins and set speed to VERY_HIGH

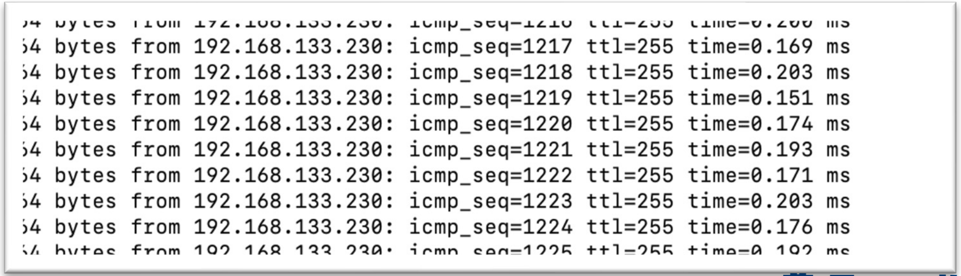
Migrate to STM32Cube FW_H7 V1.10.0 and latest CubeMX

Cleanly copy project

origin/master | origin/HEAD | Load FPGA bitstream with slave s

origin/jeisch-ethernet | jeisch-ethernet | Enable I/O Compensati

Add gain command for OFFSET DACs

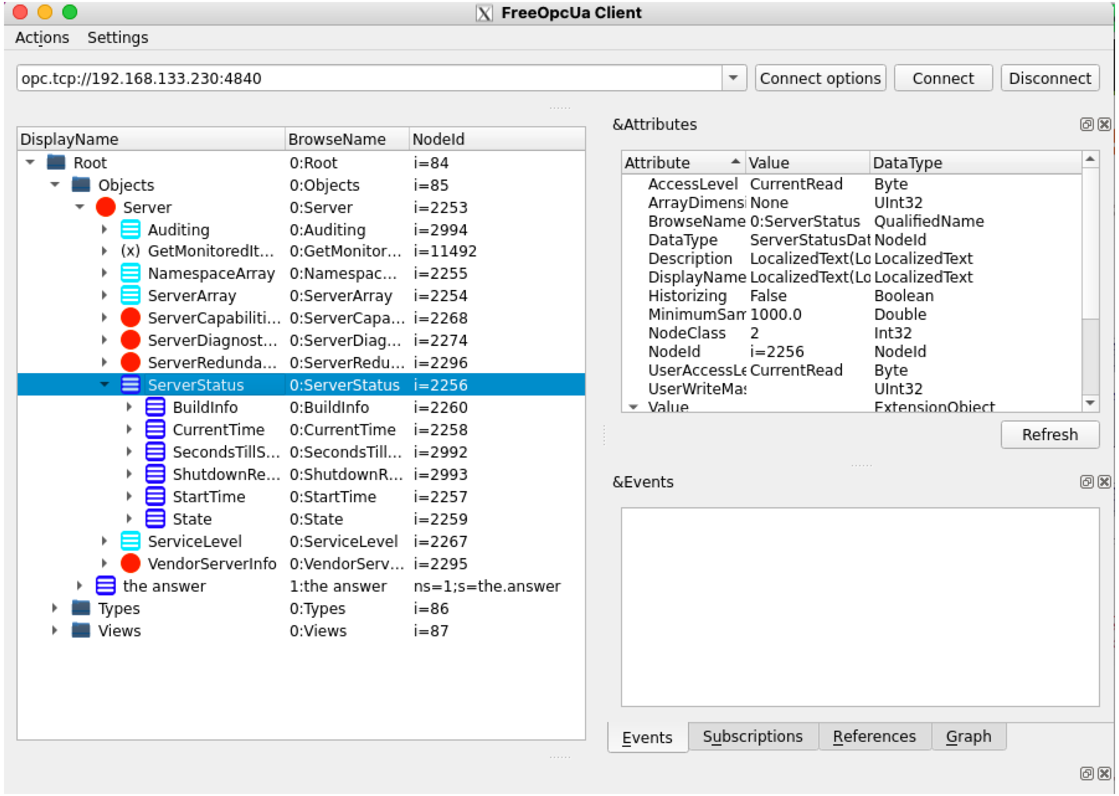


```
64 bytes from 192.168.133.230: icmp_seq=1216 ttl=255 time=0.200 ms
64 bytes from 192.168.133.230: icmp_seq=1217 ttl=255 time=0.169 ms
64 bytes from 192.168.133.230: icmp_seq=1218 ttl=255 time=0.203 ms
64 bytes from 192.168.133.230: icmp_seq=1219 ttl=255 time=0.151 ms
64 bytes from 192.168.133.230: icmp_seq=1220 ttl=255 time=0.174 ms
64 bytes from 192.168.133.230: icmp_seq=1221 ttl=255 time=0.193 ms
64 bytes from 192.168.133.230: icmp_seq=1222 ttl=255 time=0.171 ms
64 bytes from 192.168.133.230: icmp_seq=1223 ttl=255 time=0.203 ms
64 bytes from 192.168.133.230: icmp_seq=1224 ttl=255 time=0.176 ms
64 bytes from 192.168.133.230: icmp_seq=1225 ttl=255 time=0.192 ms
```

Fermilab

OPC UA Server Status – *the good news*

- The open source *open62541* version 1.2.4 has been installed and the build succeeds.
- The server source code is the same as what has been running reliably on the eval board.
- If I disable the other user tasks, the server is *sometimes* able to respond to a client request without error.



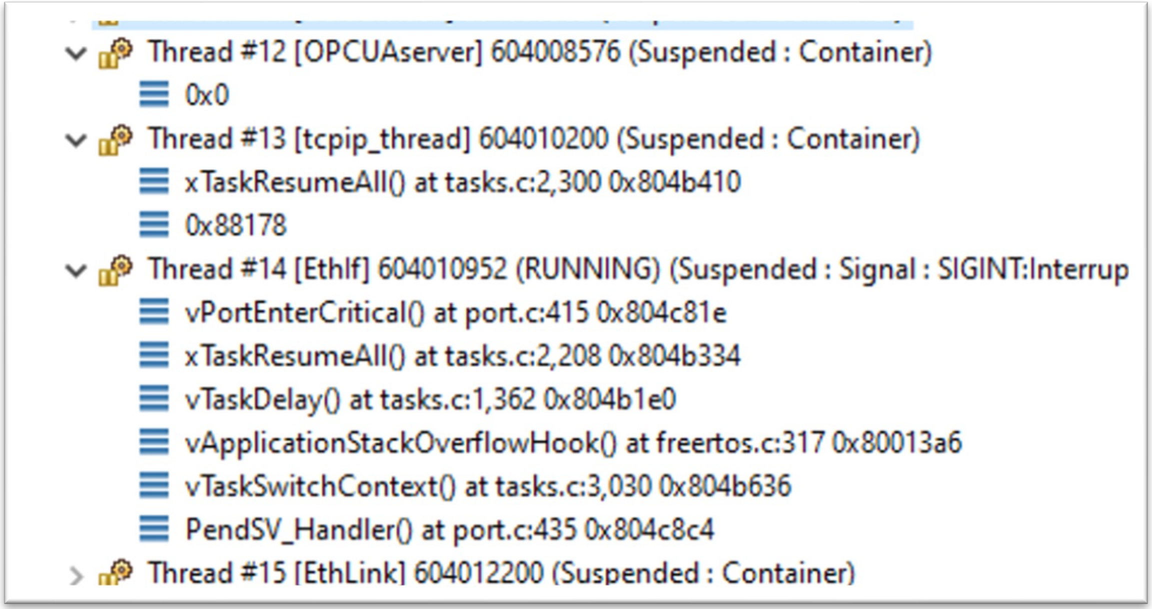
opc.tcp://192.168.133.230:4840

DisplayName	BrowseName	NodeId
Root	0:Root	i=84
Objects	0:Objects	i=85
Server	0:Server	i=2253
Auditing	0:Auditing	i=2994
(x) GetMonitoredIt...	0:GetMonitor...	i=11492
NamespaceArray	0:Namespac...	i=2255
ServerArray	0:ServerArray	i=2254
ServerCapabiliti...	0:ServerCapa...	i=2268
ServerDiagnost...	0:ServerDiag...	i=2274
ServerRedunda...	0:ServerRedu...	i=2296
ServerStatus	0:ServerStatus	i=2256
BuildInfo	0:BuildInfo	i=2260
CurrentTime	0:CurrentTime	i=2258
SecondsTillS...	0:SecondsTill...	i=2992
ShutdownRe...	0:ShutdownR...	i=2993
StartTime	0:StartTime	i=2257
State	0:State	i=2259
ServiceLevel	0:ServiceLevel	i=2267
VendorServerInfo	0:VendorServ...	i=2295
the answer	1:the answer	ns=1;s=the.answer
Types	0:Types	i=86
Views	0:Views	i=87

Attribute	Value	DataType
AccessLevel	CurrentRead	Byte
ArrayDimensi	None	UInt32
BrowseName	0:ServerStatus	QualifiedName
DataType	ServerStatusDat	NodeId
Description	LocalizedText(Lo	LocalizedText
DisplayName	LocalizedText(Lo	LocalizedText
Historizing	False	Boolean
MinimumSam	1000.0	Double
NodeClass	2	Int32
NodeId	i=2256	NodeId
UserAccessLe	CurrentRead	Byte
UserWriteMas		UInt32
Value		ExtensionObject

OPC UA Server Status – *the rest of the news*

- **Eventually, or sometimes right away, the OPCUA server and the ethernet stack have some sort of memory issue (usually identified as a stack overflow) that brings down the whole system.**
- **If the other user tasks are enabled, the memory error happens within the first few rounds of task switching.**
- **These issues happen independently of task memory allocation and heap size.**
- **Running the same code multiple times sometimes produces different results, so this is not quite deterministic.**
- **No indications at compile time that I can see.**



```
Thread #12 [OPCUAserver] 604008576 (Suspended : Container)
    0x0
Thread #13 [tcpip_thread] 604010200 (Suspended : Container)
    xTaskResumeAll() at tasks.c:2,300 0x804b410
    0x88178
Thread #14 [EthIf] 604010952 (RUNNING) (Suspended : Signal : SIGINT:Interrup
    vPortEnterCritical() at port.c:415 0x804c81e
    xTaskResumeAll() at tasks.c:2,208 0x804b334
    vTaskDelay() at tasks.c:1,362 0x804b1e0
    vApplicationStackOverflowHook() at freertos.c:317 0x80013a6
    vTaskSwitchContext() at tasks.c:3,030 0x804b636
    PendSV_Handler() at port.c:435 0x804c8c4
Thread #15 [EthLink] 604012200 (Suspended : Container)
```

Path Forward

- Any solution needs a way to have a locally stored MAC & IP addresses.
  - Who can provide that?

- Test without the OPC UA server:
  - Make a simple program to process commands from raw TCP packets.

- Test OPC UA without other DAPHNE processes:
  - Copy the current project (new Ethernet drivers/HAL) over to the eval board without the DAPHNE specific tasks and user code.

- Does anyone have any other suggestions for debugging with FreeRTOS?