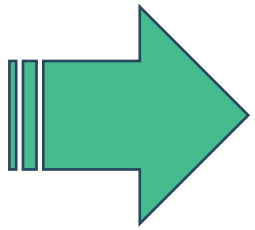
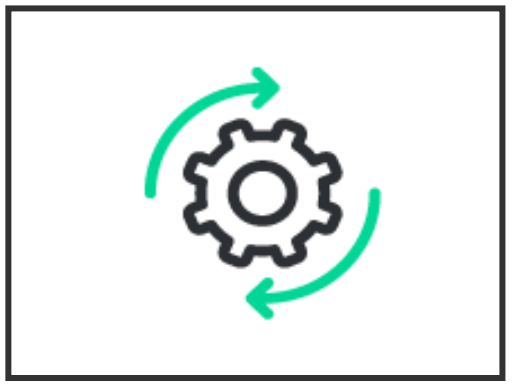
“Create your ULL FPGA application in 5 easy steps with nxFramework”

Olivier Cousin – STAC London – December 2024

exegy

Step 1: Develop your business logic





Customer design

nxFramework Business logic

Hundreds of free cores available to speed-up any new designs

Step 2: Select a board

»Wide range of FPGA boards supported

- Future-proof with no risk of board obsolescence
- Flexible to meet your specific business needs

»Improve productivity

- No need for external debugging interfaces
- Board management is integrated by default

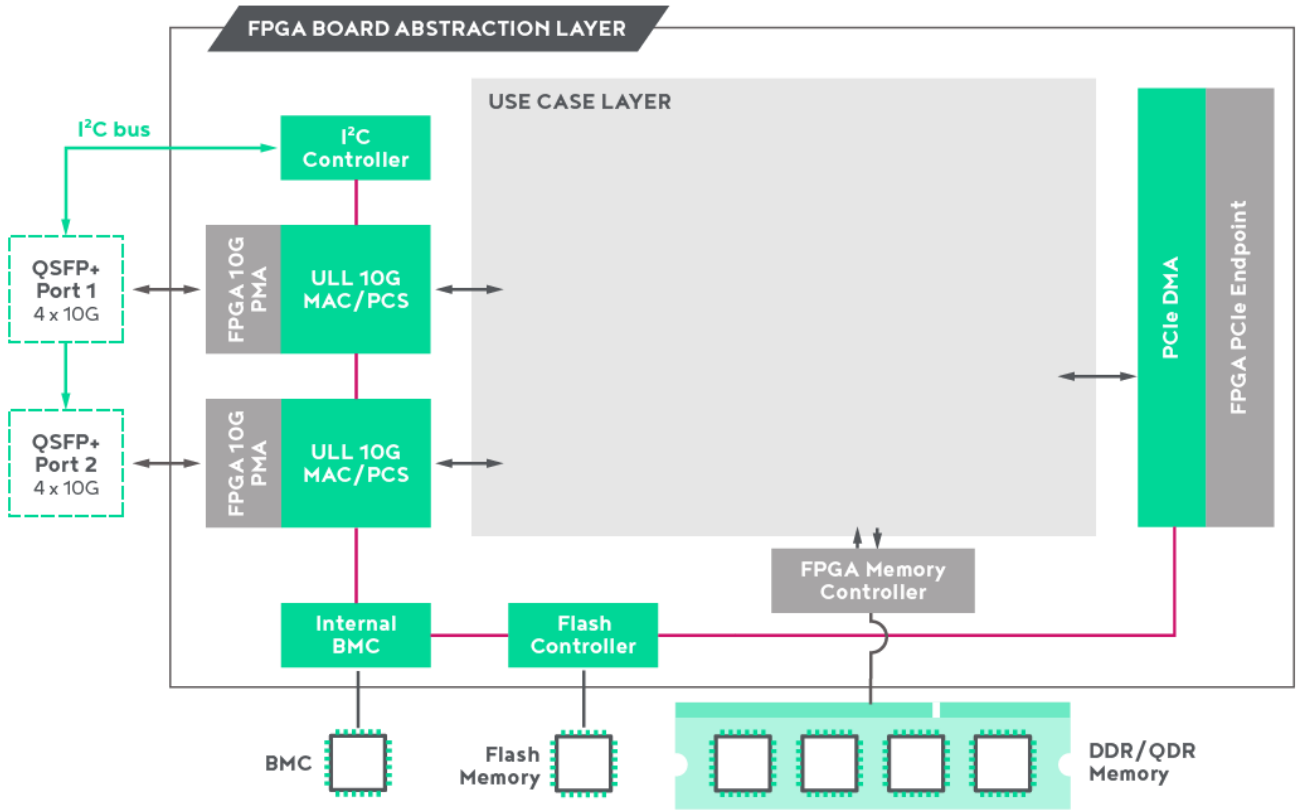
»Faster Time-to-Market

- Advanced design configuration via YAML files
- Seamless porting across different boards

Step 3: Configuration – External Interfaces

Transceivers & PHY configuration

```
################
## PHY CONFIG ##
################
QSFP0_0_EN: '1'
QSFP1_0_EN: '1'
QSFP0_MODE: 'SINGLE'
QSFP0_0_PHY_SELECT: 'PHY0'
QSFP1_MODE: 'SINGLE'
QSFP1_0_PHY_SELECT: 'PHY1'

PHY0_EN: '1'
PHY0_SPEED: '10G'
PHY0_PCS_ENYX: '1'
PHY0_PCS_RX_PIPE_COUNT: '1'
PHY0_PCS_TX_PIPE_COUNT: '1'
PHY0_MAC_DATA_WIDTH: '32'
PHY0_MAC_RX_CLK: 'PHY0_PMA_TX_CLK'
PHY0_MAC_RX_PIPE_COUNT: '1'
PHY0_MAC_TX_CLK: 'PHY0_PMA_TX_CLK'
PHY0_MAC_TX_PIPE_COUNT: '1'
PHY0_MAC_TX_PAUSE_LENGTH: '0'
```

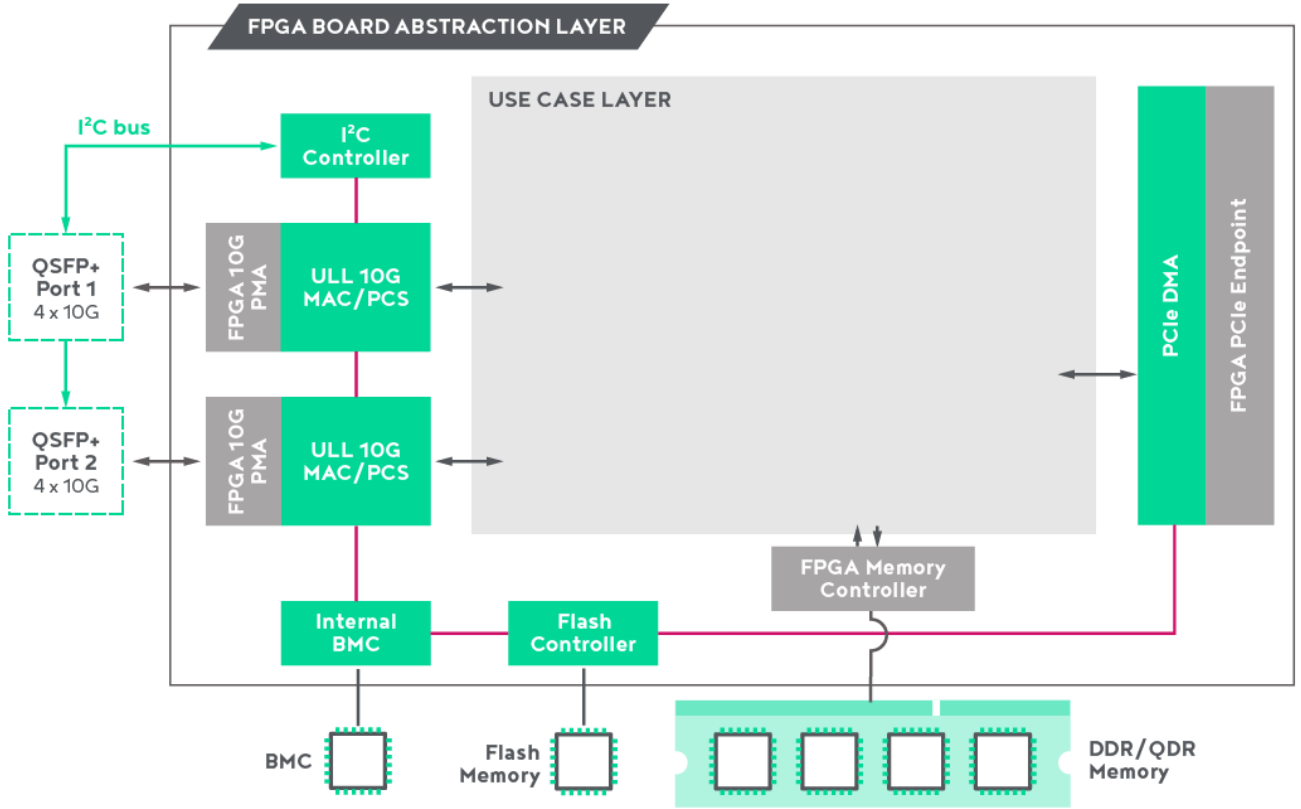


YAML File configuration, no code required

Step 3: Configuration – External Interfaces

PCIe configuration

```
################
## PCI CONFIG ##
################
PCIE_EN: '1'
PCIE_GEN: '3'
PCIE_LANE_COUNT: '8'
PCIE_DATA_WIDTH: '256'
```

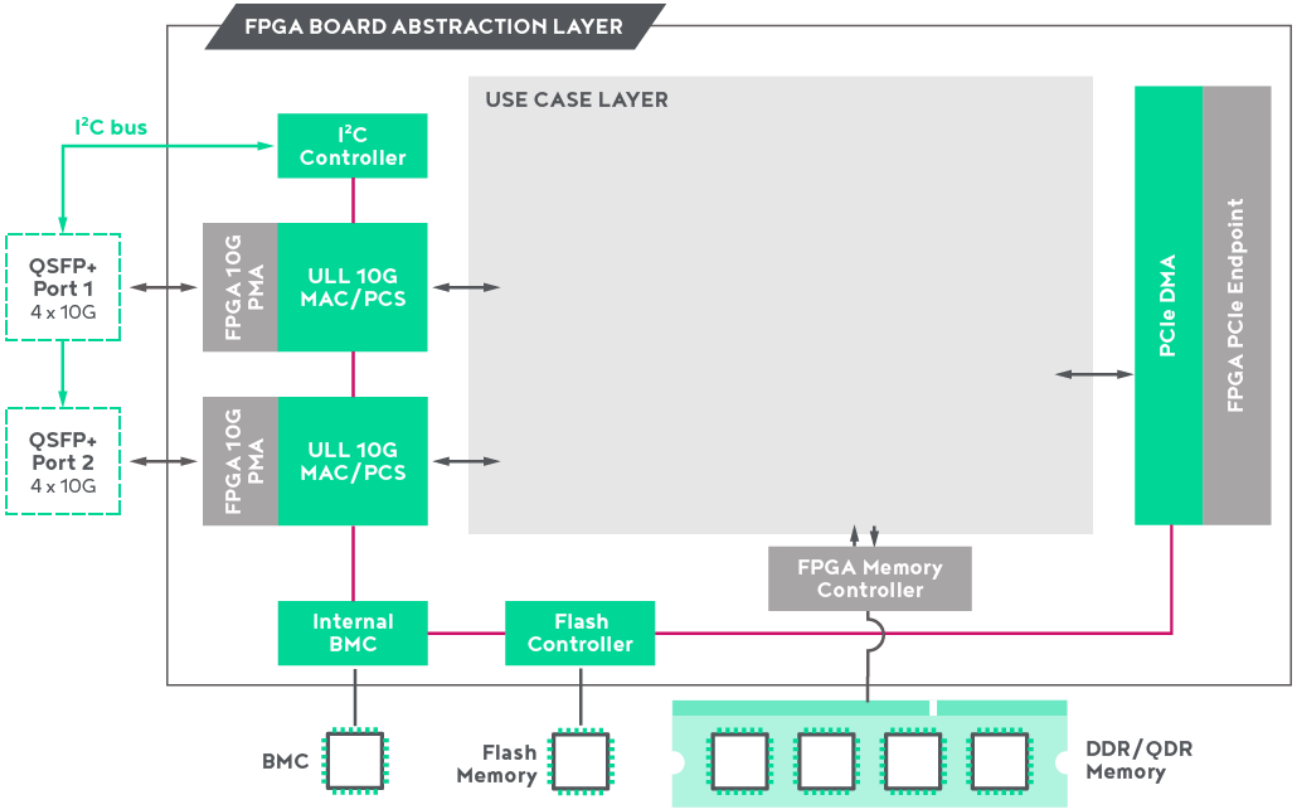


YAML File configuration, no code required

Step 3: Configuration – External Interfaces

Clk configuration

```
######################
## CUSTOM CLK CONFIG ##
######################
CUSTOM_CLK_0_EN: '1'
CUSTOM_CLK_0_FREQ_MHZ: '160.00'
```

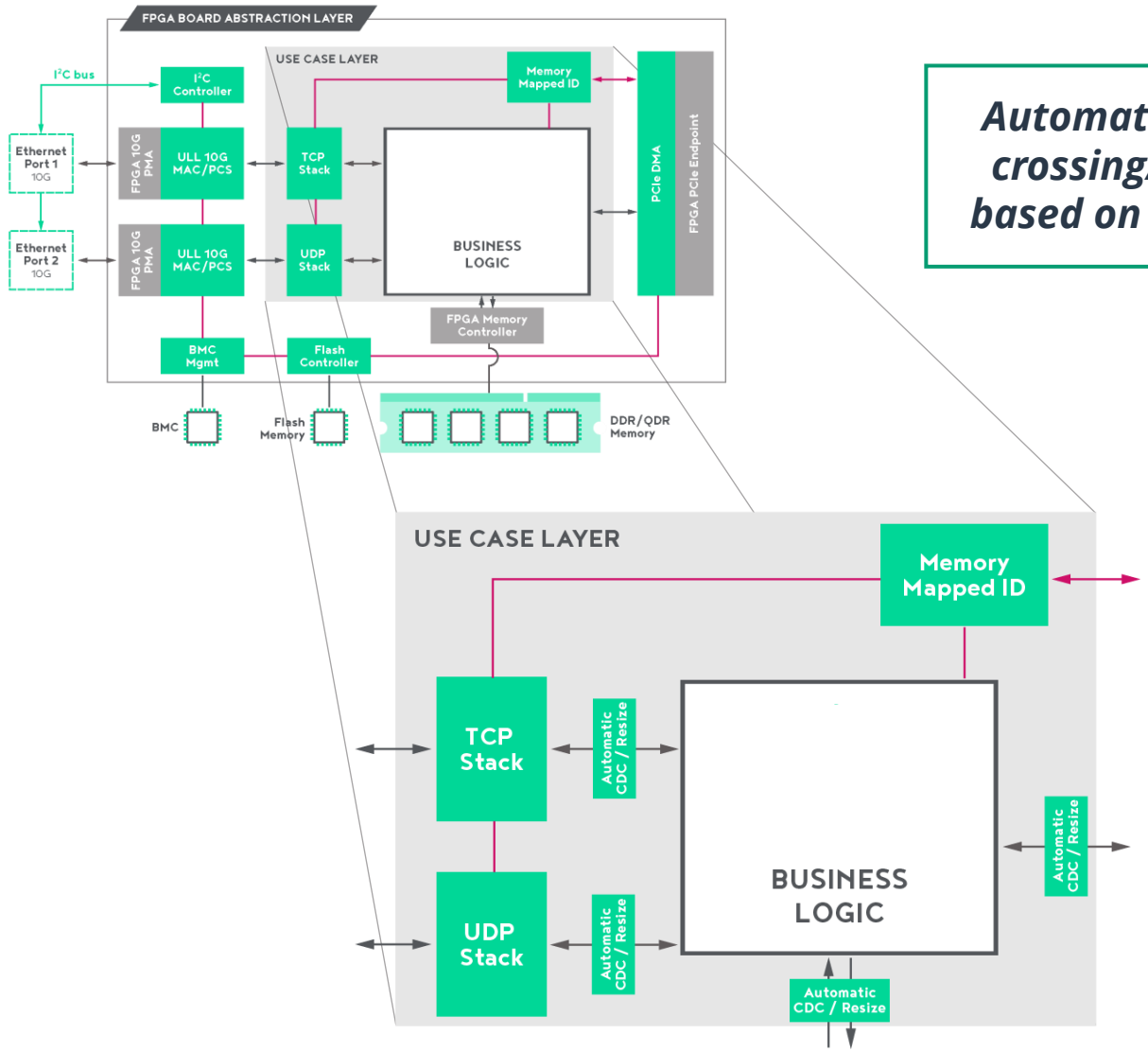


YAML File configuration, no code required

Step 3: Configuration – Application Layer

Interface configuration

```
PHY1_TCP2USER_DATA_WIDTH: '32'
PHY1_USER2TCP_DATA_WIDTH: '32'

PHY1_ULL: '1'
PHY1_MTU: '2048'
PHY1_VLAN_COUNT: '2'
PHY1_MAC_ADDRESS_COUNT: '2'
PHY1_VIRTUAL_INTERFACES_COUNT: '4'
```

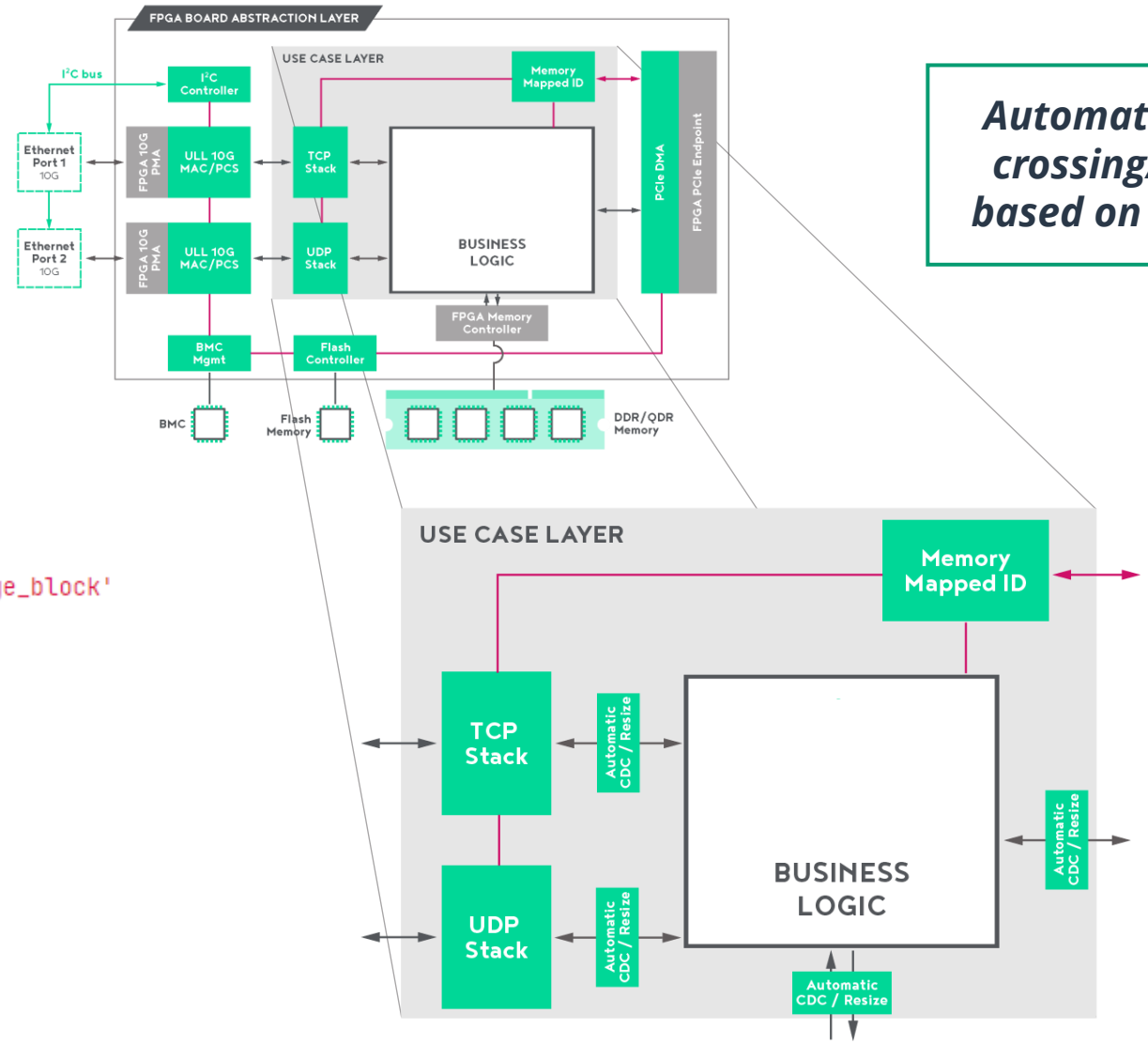


Automatic FIFO/Clock crossing/bus resizing based on configuration

Step 3: Configuration – Application Layer

TCP Stack Configuration

```
PHY1_TCP_ENABLE: '1'
PHY1_TCP_NAME: 'PROXY_SERVER'
PHY1_TCP_SESSION_COUNT: '64'
PHY1_TCP_RX_FIFO_MAX_PACKET_COUNT: '64'
PHY1_TCP_RX_FIFO_BYTE_COUNT: '4500'
PHY1_TCP_INSTANT_ACK_EN: '1'
PHY1_TCP_TX_DROP_IF_NOT_ESTABLISHED_EN: '1'
PHY1_TCP_PEER_IPV4_ADDRESS_COUNT: '16'
PHY1_TCP_RX_OOS_SEQNUM_EN: '0'
PHY1_TCP_TX_RETRANSMIT_MEM_ADDR_WIDTH: '17'
PHY1_TCP_TX_RETRANSMIT_MEM_DATA_WIDTH: '64'
PHY1_TCP_TX_RETRANSMIT_MEM_CLK: 'CUSTOM_CLK_0'
PHY1_TCP_TX_RETRANSMIT_MEM_INTERNAL_RAM_TYPE: 'large_block'
PHY1_TCP_TX_RETRANSMIT_MEM_INTERNAL_LATENCY: '10'
PHY1_TCP_EMI_STATUS_EN: '0'
PHY1_TCP_EMI_CREDIT_EN: '0'
PHY1_TCP_RX_OUTPUT_PIPE_COUNT: '1'
PHY1_TCP_FORCE_M20K_EN: '0'
```

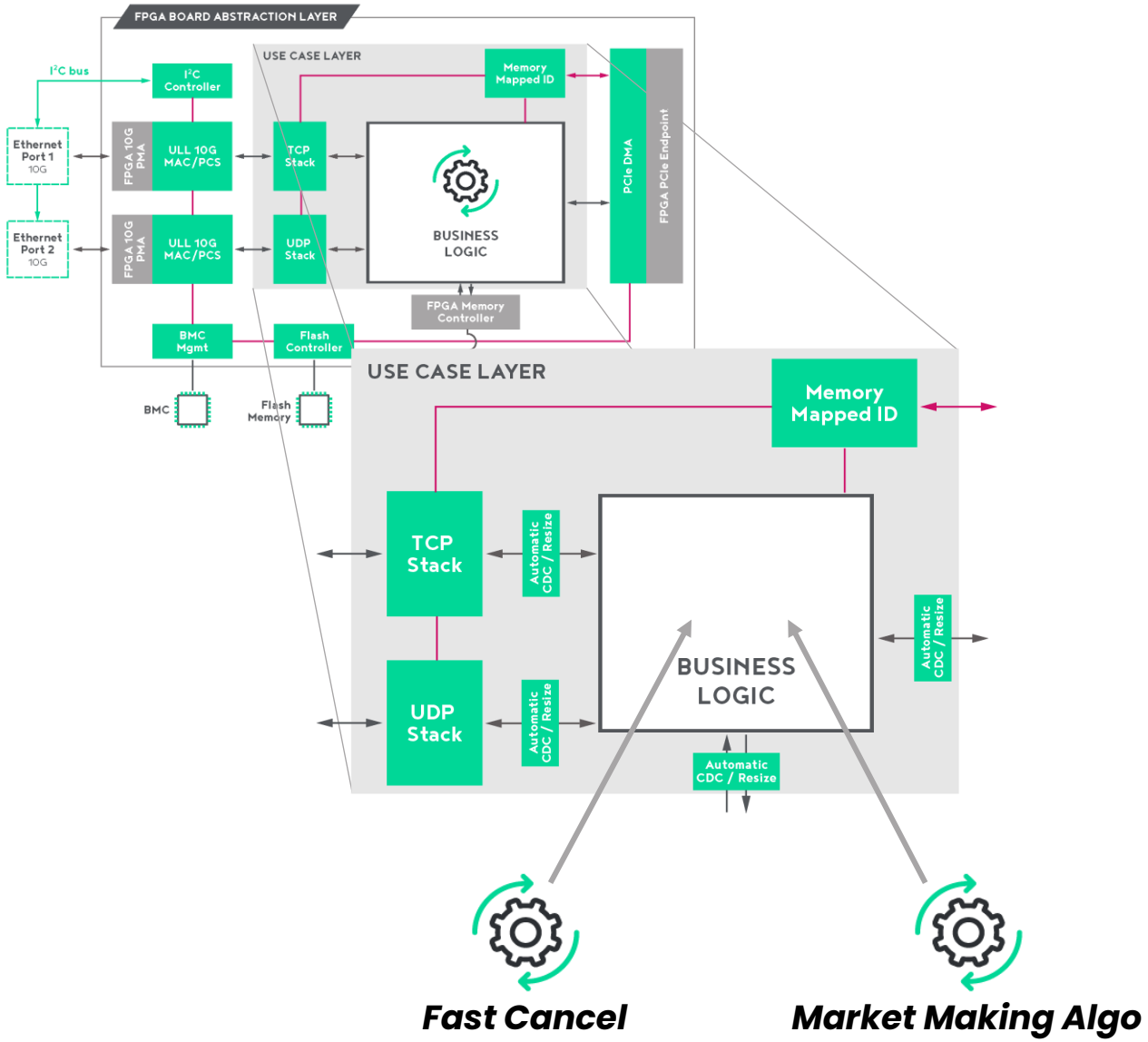


Automatic FIFO/Clock crossing/bus resizing based on configuration

Step 3: Configuration – User Sandbox

TCP Interface

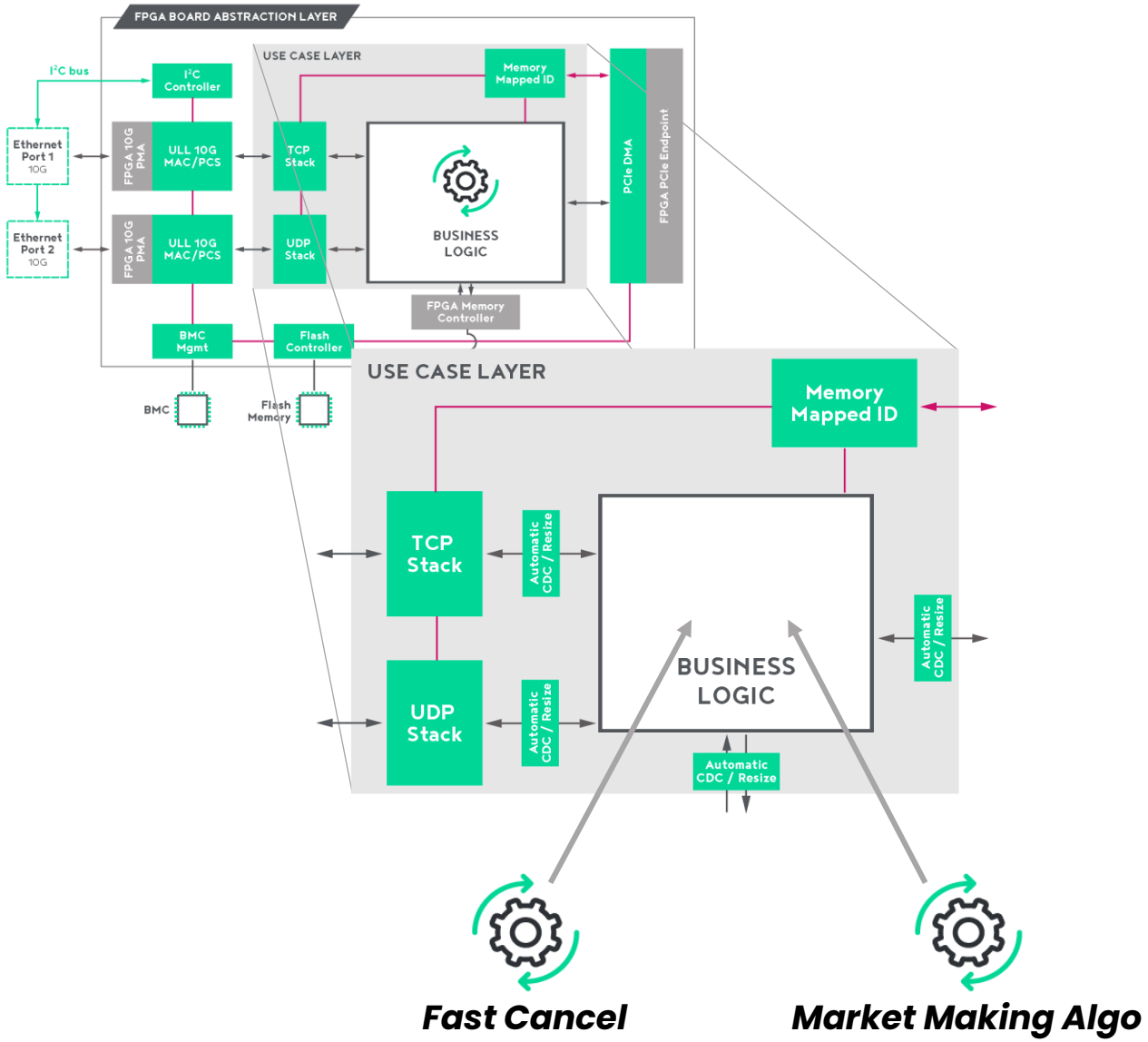
```
PHY_TCP_SB_ST_IN_DATA_WIDTH: '32'

PHY0_TCP_SB_ST_IN_CLK: 'PHY0_PMA_TX_CLK'
PHY0_TCP_TO_SB_FIFO_EN: '0'
PHY0_TCP_TO_SB_FIFO_STORE_FW: '0'
PHY0_TCP_TO_SB_FIFO_CLEAN_DROP: '0'
PHY0_TCP_TO_SB_PIPE_COUNT: '0'

PHY1_TCP_SB_ST_IN_CLK: 'PHY0_PMA_TX_CLK'
PHY1_TCP_TO_SB_FIFO_EN: '0'
PHY1_TCP_TO_SB_FIFO_STORE_FW: '0'
PHY1_TCP_TO_SB_FIFO_CLEAN_DROP: '0'
PHY1_TCP_TO_SB_PIPE_COUNT: '0'

PHY_TCP_SB_ST_OUT_DATA_WIDTH: '32'

PHY0_TCP_SB_ST_OUT_CLK: 'PHY0_PMA_TX_CLK'
PHY0_TCP_FROM_SB_FIFO_EN: '0'
PHY0_TCP_FROM_SB_FIFO_STORE_FW: '0'
PHY0_TCP_FROM_SB_FIFO_CLEAN_DROP: '0'
PHY0_TCP_FROM_SB_PIPE_COUNT: '0'

PHY1_TCP_SB_ST_OUT_CLK: 'PHY0_PMA_TX_CLK'
PHY1_TCP_FROM_SB_FIFO_EN: '0'
PHY1_TCP_FROM_SB_FIFO_STORE_FW: '0'
PHY1_TCP_FROM_SB_FIFO_CLEAN_DROP: '0'
PHY1_TCP_FROM_SB_PIPE_COUNT: '0'
```



Step 3: Configuration – User Sandbox

DMA Interface

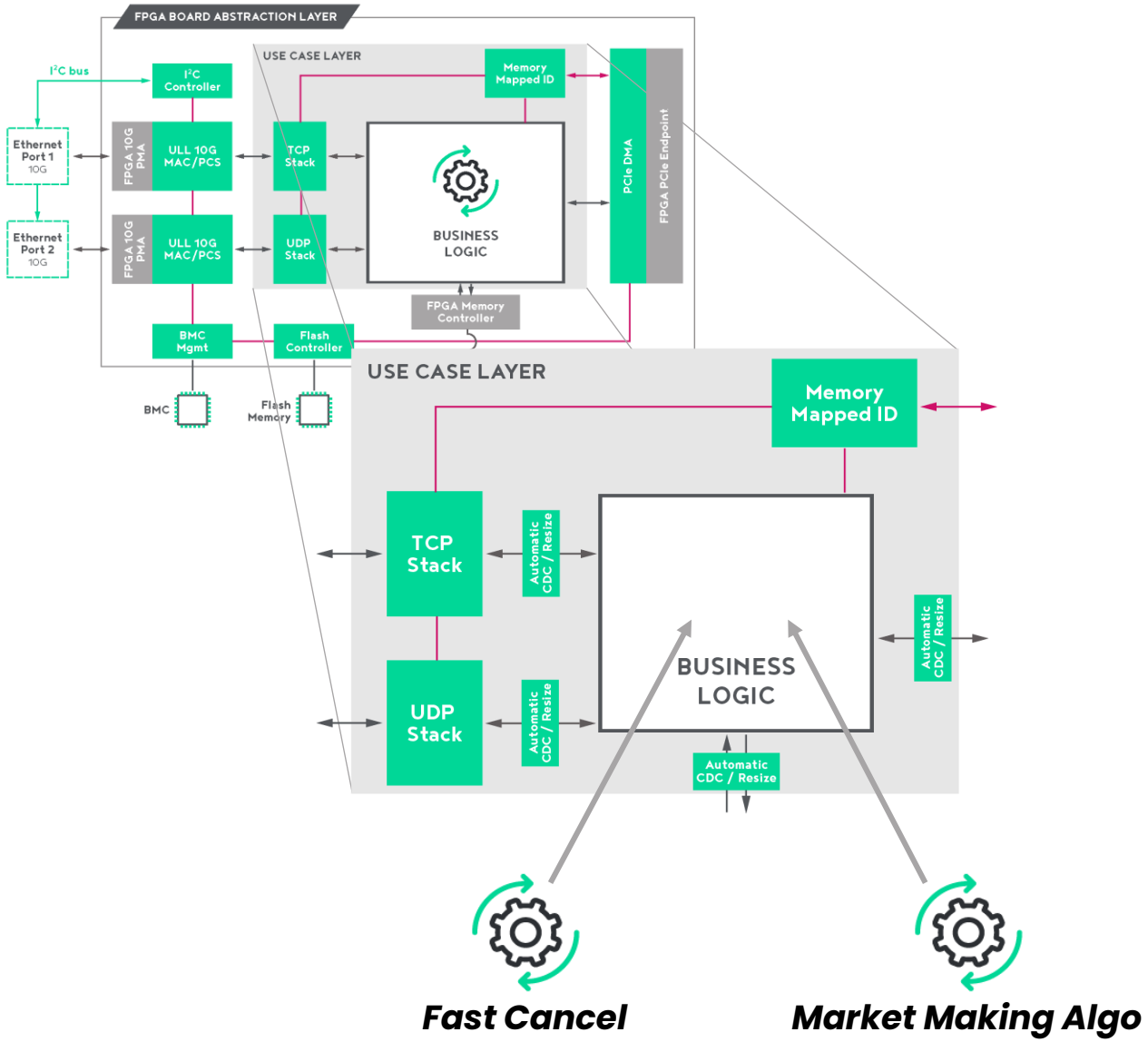
```
DMA_SB_ST_IN_DATA_WIDTH: '256'
DMA0_SB_ST_IN_CLK: 'PCIE_USER_CLK'
DMA0_TO_SB_PIPE_COUNT: '2'

DMA_SB_ST_OUT_DATA_WIDTH: '256'
DMA0_SB_ST_OUT_CLK: 'PCIE_USER_CLK'
DMA0_FROM_SB_PIPE_COUNT: '2'
```



Step 3: Configuration – User Sandbox

MM Interface

```
SANDBOX_MM_CLK: 'INTERNAL_50_CLK'
```



Step 4: Simulation & Firmware Generation

- nxFramework is built around a Cmake workflow for compatibility with existing orchestration tools

- Targets are automatically generated based on the configuration
  - **Design simulation:**

    ```
    make sim_nxuser_sandbox_tick_to_trade
    ```

  - **Firmware generation:**

    ```
    make firmware_ul3524-vu2p_tick_to_trade
    ```

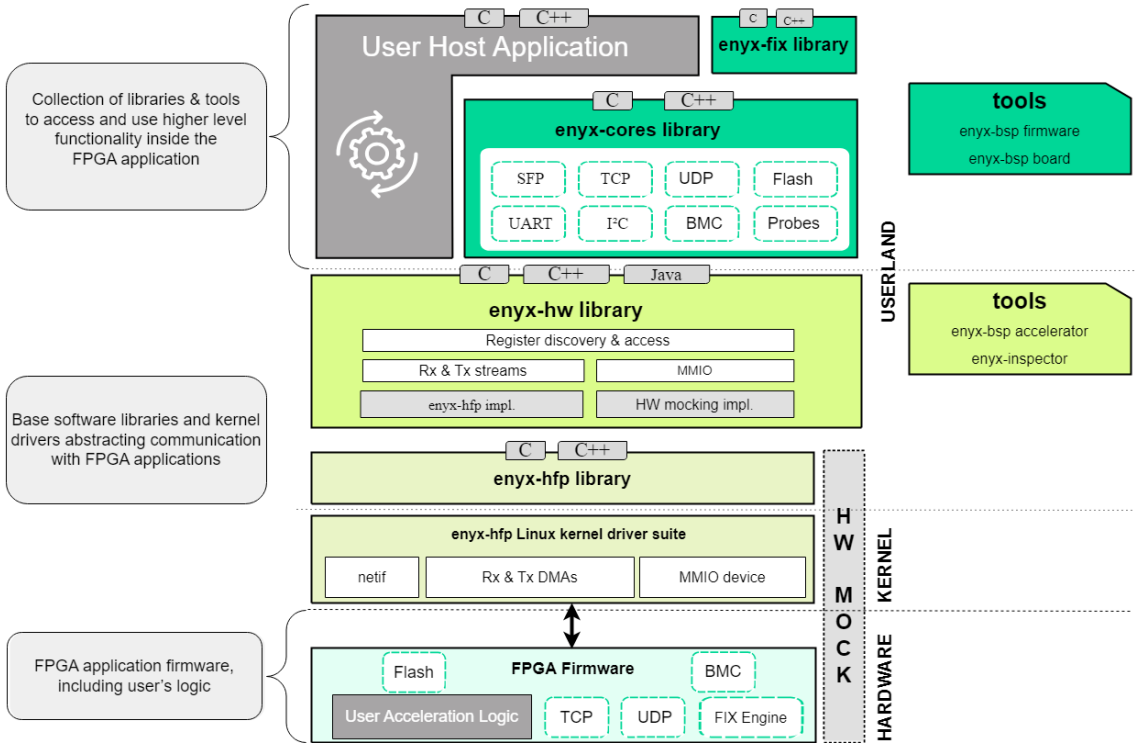
Step 5: Deploy, Monitor, Configure, Debug

» Board Support Package

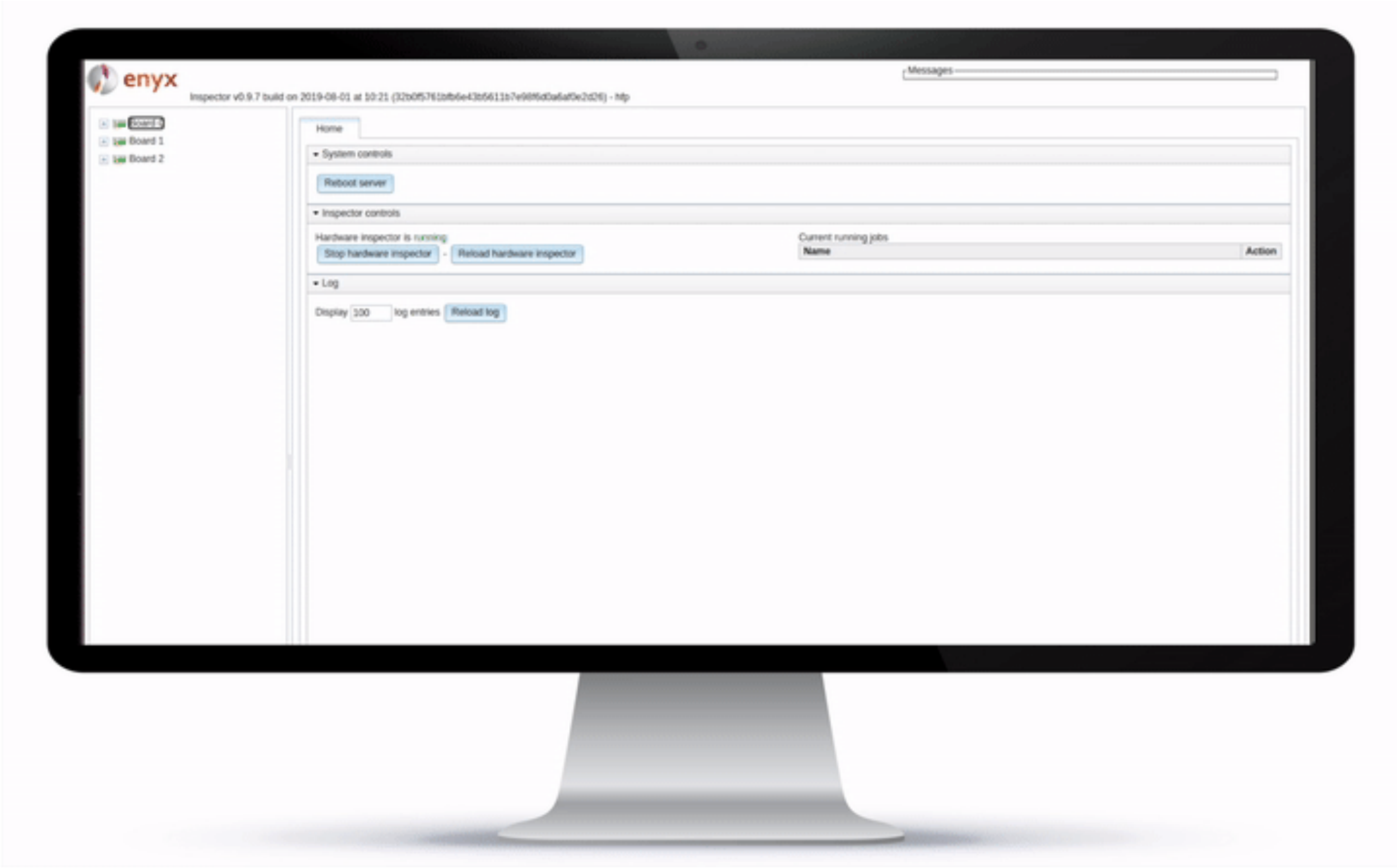
- Board monitoring & configuration
- PCIe DMA channel monitoring
- Register read & write commands
- QSFP configuration & status
- Memory testing
- System statistics reporting

» Software API for Debug

- Network stack configuration & statistics
- Low level IP management
- Software API for hardware emulation



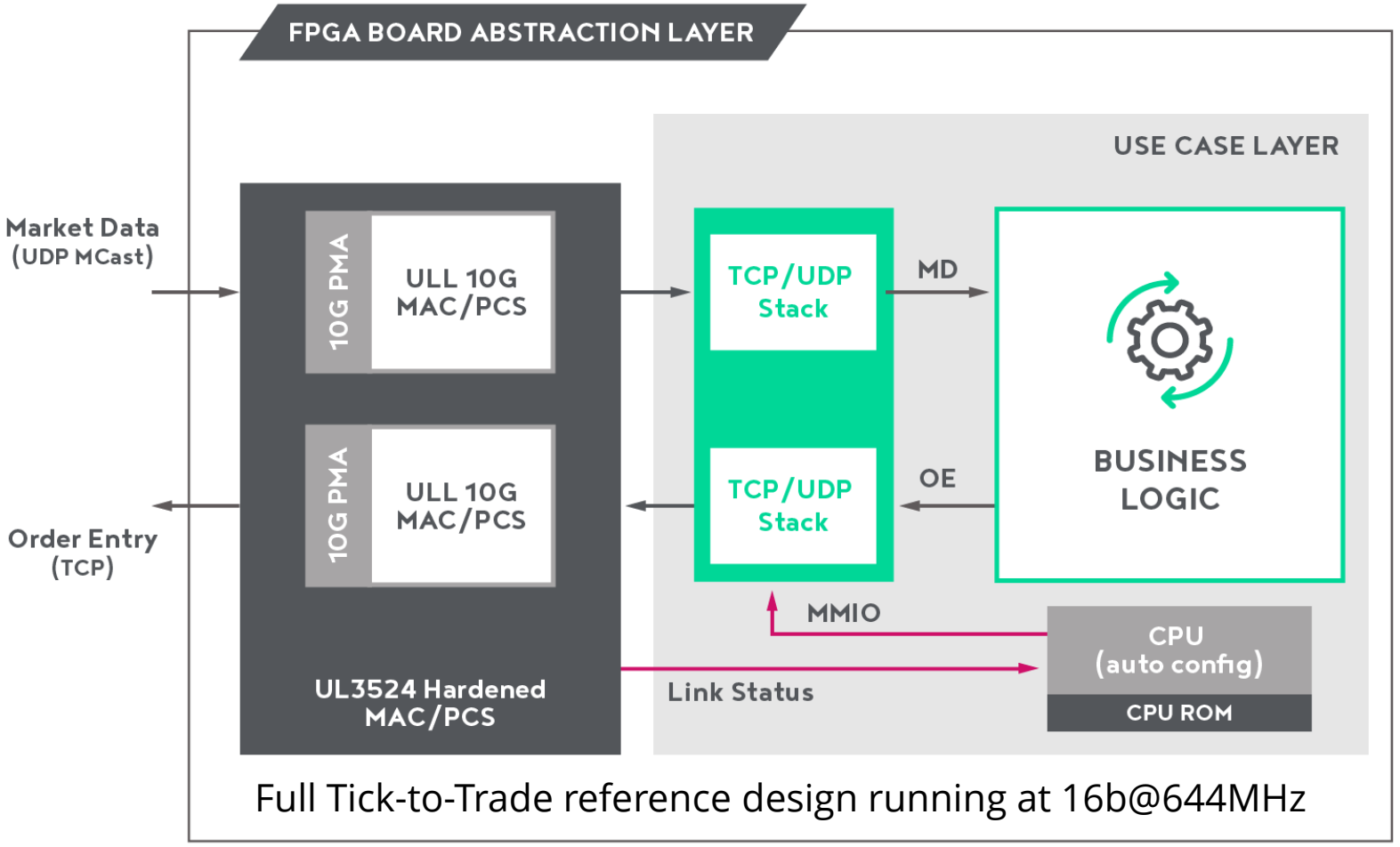
Step 5: Deploy, Monitor, Configure, Debug



exegy

nxFramework: *Tick-to-Trade Record*

13.894 ns = Exegy T2T + GTF + Serialization + Preamble (6.4ns) + QSFP & PCB traces*



Tick-to-Trade design now available on the UL3422

*STAC-T0 latency on the AMD UL3524

Seamless Migration To A New Platform

CMakeLists.txt 182 bytes

```
1  cmake_minimum_required(VERSION 3.14)
2  project(nxframework-SRD LANGUAGES NONE)
3
4  find_package(nxFramework-fdk-Vanilla-xupvv8 REQUIRED)
5
6  add_subdirectory(cores)
7  add_subdirectory(config)
8
```

CMakeLists.txt 182 bytes

```
1  cmake_minimum_required(VERSION 3.14)
2  project(nxframework-SRD LANGUAGES NONE)
3
4  find_package(nxFramework-fdk-Vanilla-ul3422 REQUIRED)
5
6  add_subdirectory(cores)
7  add_subdirectory(config)
8
```

One CMake variable change