Can you let me know board, performance of chip, code snipchet of using in MCU like ARM?

The **LN-3000 series boards** used in Johnson Controls automation networks integrate both **Neuron 3150 chips** (for LonWorks protocol handling) and **ARM Cortex-M3 MCUs** (for application control). These two cores work together: the Neuron handles LON network messaging while the ARM executes control logic and sensor/actuator management.

Board and Processor Architecture

- **Controller examples**: LN-PRG203, LN-PRG600, LN-IO modules.
- **Main MCU**: **STM32 (ARM Cortex-M3, 32-bit, 64 MHz)** with 64 KB flash and 20 KB RAM for user applications and I/O logic.[1]
- **Network processor**: **Echelon Neuron 3150, 8-bit, 10 MHz**, with integrated LONWORKS protocol stack, 64 KB flash, 128 KB storage memory.[2] [3]
- **Board power**: 24 VAC/DC ±15%, typically 6–25 VA; devices are UL916, CE, FCC approved.[2] [1]
- **Communication**: LONWORKS TP/FT-10 bus at 78 kbps (peer-to-peer); Neuron I/O pins link to ARM MCU via SPI or UART; status LEDs indicate TX/RX and service state.[4] [2]
- **ADC/DAC**: 16-bit ADC, 12-bit DAC configurable for PWM 2 s–15 min duty cycles.

Performance and Design Notes

- The **ARM Cortex-M3** provides deterministic 32-bit real-time control, suitable for PID loops, filtering, or smart I/O expansion tasks.
- The **Neuron 3150's MIP** (Microprocessor Interface Program) offloads LON communication, freeing MCU cycles for application logic.
- The **10 MHz external crystal** yields a **5 MHz internal system clock**, with network throughput optimized for 78 kbps traffic frames.[5] [4]
- Combined, this configuration delivers precise synchronization between digital (DI), analog (AO), and field communication (DL) layers of the LN-3000 distributed system.

Example: ARM Integration Code Snippet

Below is a minimal conceptual example (C for STM32 MCU) illustrating how an embedded ARM Cortex-M3 might interface with the Neuron 3150 through UART (serial link) to transmit LON frames:

```
#include "stm32f10x.h"

#define LON_UART USART1

void LON_UART_Init(void) {
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_USART1, ENABLE);
    USART_InitTypeDef uart;
    uart.USART_BaudRate = 78125;           // Compatible with TP/FT-10 bit rate
    uart.USART_WordLength = USART_WordLength_8b;
    uart.USART_StopBits = USART_StopBits_1;
    uart.USART_Parity = USART_Parity_No;
    uart.USART_Mode = USART_Mode_Tx | USART_Mode_Rx;
    uart.USART_HardwareFlowControl = USART_HardwareFlowControl_None;
    USART_Init(LON_UART, &uart);
    USART_Cmd(LON_UART, ENABLE);
}

void LON_SendFrame(uint8_t *data, uint16_t size) {
    for (uint16_t i = 0; i < size; i++) {
        while (USART_GetFlagStatus(LON_UART, USART_FLAG_TXE) == RESET);
        USART_SendData(LON_UART, data[i]);
    }
}

int main(void) {
    LON_UART_Init();
    uint8_t lon_msg[] = {0x80, 0x01, 0x23, 0x45, 0x67};  // Example frame payload
    while (1) {
        LON_SendFrame(lon_msg, sizeof(lon_msg));
        for (volatile int delay = 0; delay < 50000; delay++);
    }
}
```

Explanation

- The example sets a **78125 baud UART** to align with the LON TP/FT-10 physical layer rate.
- The MCU pushes payloads via UART/SPI into the Neuron 3150's TX buffer (the MIP handles framing, CRC, and network token control).
- Received frames (RX) can trigger interrupts for application handling, feeding AO/DI updates back into the LN distributed logic.

In practice, firmware for Johnson Controls LN-3000 boards uses a **Neuron C stack** on the Neuron 3150 and **C/C++ or IEC 61131 logic** on the ARM side, linked over SPI/UART channels. Developers compile Neuron C code with Echelon toolchains (e.g., NodeBuilder or ShortStack FX) and ARM firmware with standard STM32 toolchains. [6] [7]

⁂

1. https://docs.johnsoncontrols.com/bas/api/khub/documents/f3pM173QRRZxVr68sQEwrQ/content
2. https://docs.johnsoncontrols.com/bas/api/khub/documents/3~nNV0beoEtRCIcCslgCbA/content
3. http://www.stitcs.com/en/53011e.html

4. https://techstep.co.nz/technologies/lonworks/
5. https://www.mouser.com/datasheet/2/713/14305R_Neuron_5000_Processor-2486418.pdf
6. http://downloads.echelon.com/support/documentation/manuals/devtools/078-0366-01B_ShortStack_ARM7PortUG.pdf
7. https://www.renesas.com/en/document/mat/neuron-c-reference-guide
8. https://docs.johnsoncontrols.com/bas/api/khub/documents/PqKgF0_xN6VxPaf~70FAkw/content
9. https://docs.johnsoncontrols.com/bas/api/khub/documents/Im5MeB_Ilq2avMzvwAS9vg/content
10. https://docs.johnsoncontrols.com/bas/api/khub/documents/dJXYigcLUej5MWdhAM8SiQ/content
11. http://www.loncontrol.com/Neuron_TRM.pdf
12. https://www.infineon.com/dgdl/Infineon-CY7C53150_CY7C53120_Neuron_Chip_Network_Processor-DataSheet-v13_00-EN.pdf?fileId=8ac78c8c7d0d8da4017d0eca599a4309
13. https://docs.johnsoncontrols.com/bas/api/khub/documents/YGBxgvHT~P8fizIKkksmRg/content
14. https://blog.mbedded.ninja/misc/lon-ansi709-2-communication-protocol/Echelon-005-0199-01B-Series-5000-Chip-Data-Book.pdf
15. https://blog.mbedded.ninja/electronics/communication-protocols/lon-ansi709-2-communication-protocol/
16. https://docs.johnsoncontrols.com/bas/api/khub/documents/YxJxxJTQmHQFPjqwSsKg3g/content
17. https://www.renesas.cn/zh/document/mat/neuron-c-reference-guide
18. https://www.scss.tcd.ie/SCSSTreasuresCatalog/hardware/TCD-SCSS-T.20230524.001/Toshiba-Neuron-TMPN3120-TMPN3150-Processors.pdf
19. https://docs.johnsoncontrols.com/bas/api/khub/documents/_cDx99GMs8zZBkiE4RWwuQ/content
20. https://docs.johnsoncontrols.com/bas/api/khub/documents/Yg0Qaa_CHMo1c8ug5wYhfw/content