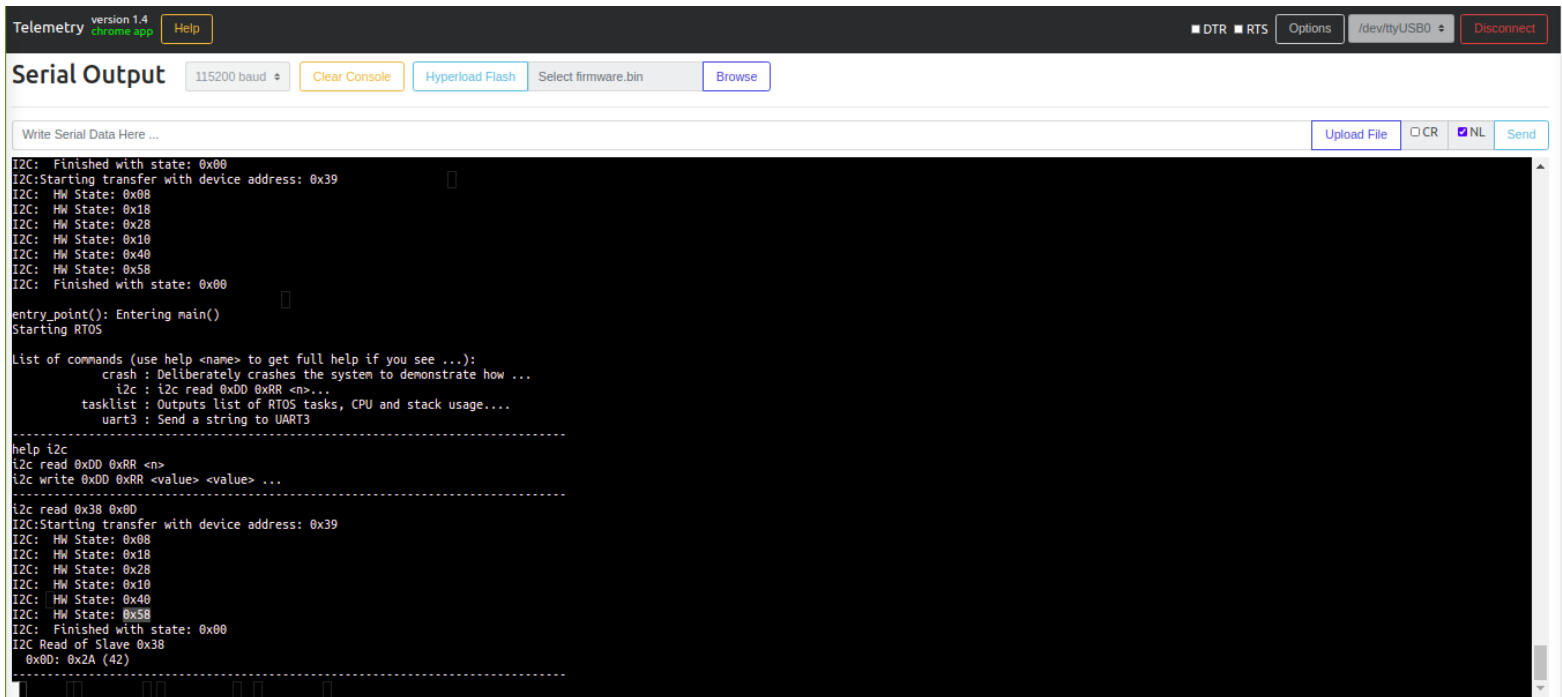


1) To read WHO_AM_I register:

- Sent “i2c read 0x38 0x0D” CLI command
- Below the Telemetry screenshot of same

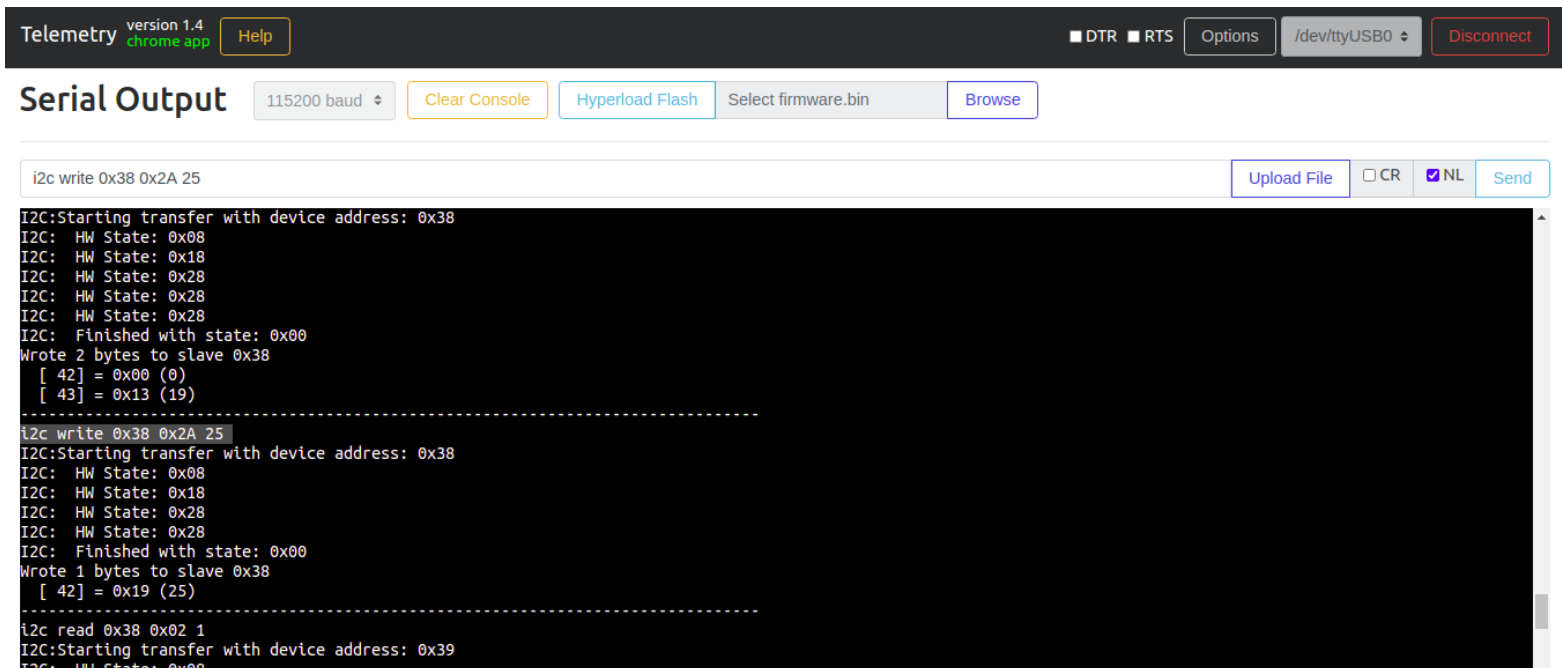


The screenshot shows the Telemetry interface with the Serial Output window open. The baud rate is set to 115200. The console displays the following text:

```
Telemetry version 1.4 chrome app Help
Serial Output 115200 baud Clear Console Hyperload Flash Select firmware.bin Browse
Write Serial Data Here ... Upload File CR NL Send
I2C: Finished with state: 0x00
I2C: Starting transfer with device address: 0x39
I2C: HW State: 0x08
I2C: HW State: 0x18
I2C: HW State: 0x28
I2C: HW State: 0x10
I2C: HW State: 0x40
I2C: HW State: 0x58
I2C: Finished with state: 0x00
entry_point(): Entering main()
Starting RTOS
List of commands (use help <name> to get full help if you see ...):
  crash : Deliberately crashes the system to demonstrate how ...
  i2c : i2c read 0x0D 0xRR <n>...
  tasklist : Outputs list of RTOS tasks, CPU and stack usage...
  uart3 : Send a string to UART3
-----
help i2c
i2c read 0x0D 0xRR <n>
i2c write 0x0D 0xRR <value> <value> ...
-----
i2c read 0x38 0x0D
I2C: Starting transfer with device address: 0x39
I2C: HW State: 0x08
I2C: HW State: 0x18
I2C: HW State: 0x28
I2C: HW State: 0x10
I2C: HW State: 0x40
I2C: HW State: 0x58
I2C: Finished with state: 0x00
I2C Read of Slave 0x38
0x0D: 0x2A (42)
```

2) To write to CTRL_REG1 register of Accelerometer sensor:

- Sent “i2c write 0x38 0x2A 25” CLI command



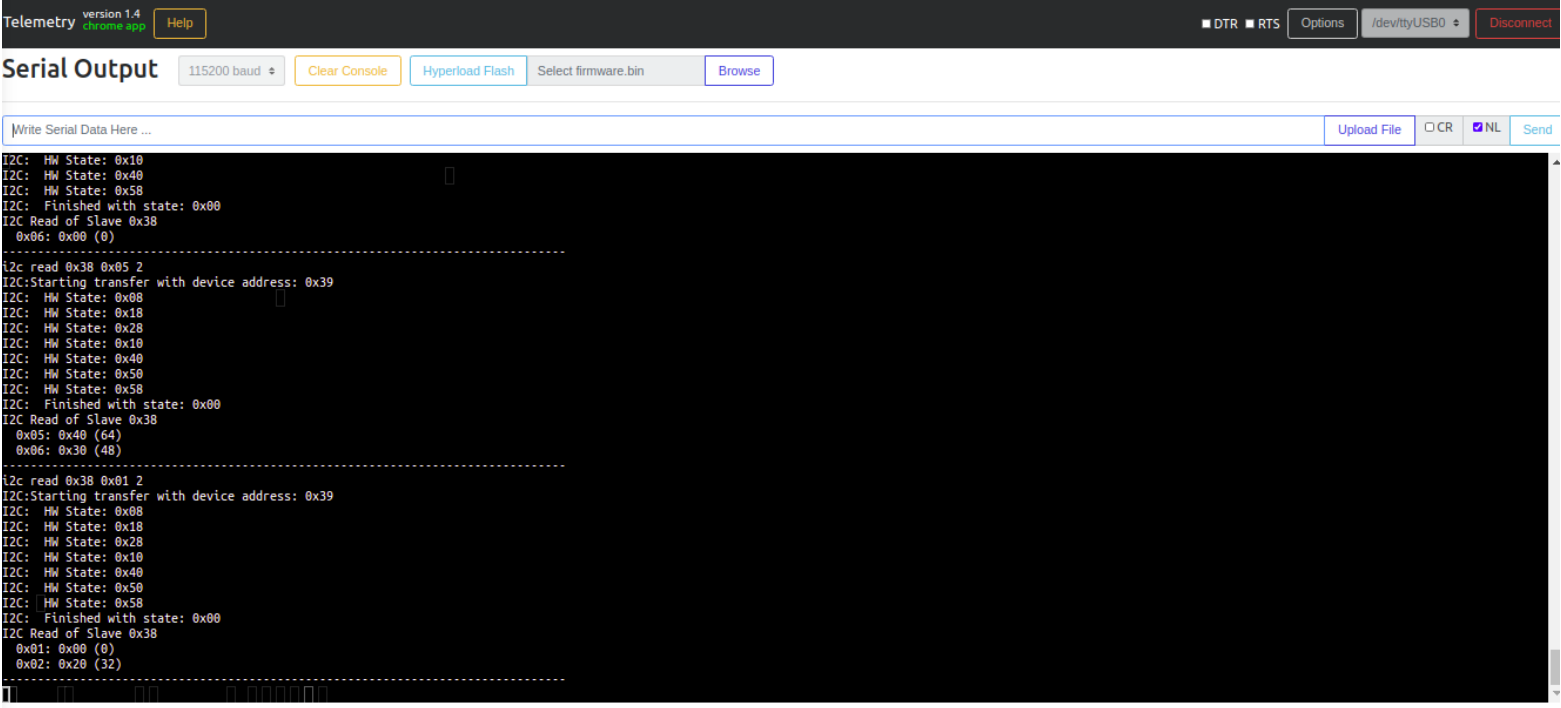
The screenshot shows the Telemetry interface with the Serial Output window open. The baud rate is set to 115200. The console displays the following text:

```
Telemetry version 1.4 chrome app Help
Serial Output 115200 baud Clear Console Hyperload Flash Select firmware.bin Browse
i2c write 0x38 0x2A 25 Upload File CR NL Send
I2C: Starting transfer with device address: 0x38
I2C: HW State: 0x08
I2C: HW State: 0x18
I2C: HW State: 0x28
I2C: HW State: 0x28
I2C: HW State: 0x28
I2C: Finished with state: 0x00
Wrote 2 bytes to slave 0x38
[ 42] = 0x00 (0)
[ 43] = 0x13 (19)
-----
i2c write 0x38 0x2A 25
I2C: Starting transfer with device address: 0x38
I2C: HW State: 0x08
I2C: HW State: 0x18
I2C: HW State: 0x28
I2C: HW State: 0x28
I2C: Finished with state: 0x00
Wrote 1 bytes to slave 0x38
[ 42] = 0x19 (25)
-----
i2c read 0x38 0x02 1
I2C: Starting transfer with device address: 0x39
I2C: HW State: 0x08
```

3) To read X and Z axis value of accelerometer:

→ sent “i2c read 0x38 0x01 2” CLI command to read X axis data

→ sent “i2c read 0x38 0x05 2” CLI command to read Z axis data



The screenshot shows the Telemetry version 1.4 chrome app interface. The top bar includes a 'Help' button, status indicators for DTR and RTS, an 'Options' button, a dropdown menu for '/dev/ttyUSB0', and a 'Disconnect' button. Below the top bar, the 'Serial Output' section features a baud rate of 115200, buttons for 'Clear Console', 'Hyperload Flash', and 'Browse', and a 'Select firmware.bin' button. The main area is a terminal window with a 'Write Serial Data Here ...' input field and buttons for 'Upload File', 'CR', 'NL', and 'Send'. The terminal displays the following I2C communication logs:

```
I2C: HW State: 0x10
I2C: HW State: 0x40
I2C: HW State: 0x58
I2C: Finished with state: 0x00
I2C Read of Slave 0x38
  0x06: 0x00 (0)
-----
i2c read 0x38 0x05 2
I2C:Starting transfer with device address: 0x39
I2C: HW State: 0x08
I2C: HW State: 0x18
I2C: HW State: 0x28
I2C: HW State: 0x10
I2C: HW State: 0x40
I2C: HW State: 0x50
I2C: HW State: 0x58
I2C: Finished with state: 0x00
I2C Read of Slave 0x38
  0x05: 0x40 (64)
  0x06: 0x30 (48)
-----
i2c read 0x38 0x01 2
I2C:Starting transfer with device address: 0x39
I2C: HW State: 0x08
I2C: HW State: 0x18
I2C: HW State: 0x28
I2C: HW State: 0x10
I2C: HW State: 0x40
I2C: HW State: 0x50
I2C: HW State: 0x58
I2C: Finished with state: 0x00
I2C Read of Slave 0x38
  0x01: 0x00 (0)
  0x02: 0x20 (32)
-----
```