I2C in Linux

⇒ I^2C bus is generally used to connect relatively low-Speed sensors.

⇒ I^2C is extreamly popular due to its ease of use and ability to control multiple peripherals while utilizing only two pins on the host controller.

i^2C tools

- → Set of i^2c program that makes it easy to debug i^2c devices without having to write any code.

  Example

  `$ i2cdetect -r 2`

  → Will send read commands on the /dev/i2c-2 line to Probe for address, and return any device found.

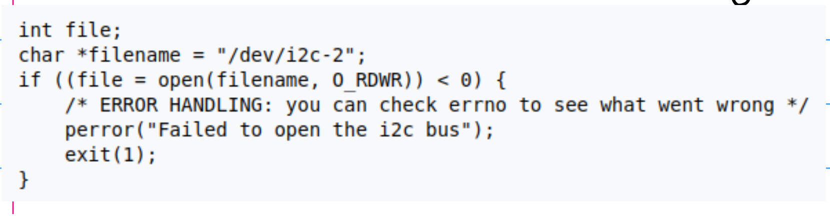
- → I2Cget and I2Cset write and read to devices respectively.

⇒ Basic I2C bus control can be accomplished with:

- → Open
- → iocl
- → read
- → write

★ Opening the bus

⇒ Open returns a new file descriptor (non negative integer) which can be then used to configure the bus.



```
int file;
char *filename = "/dev/i2c-2";
if ((file = open(filename, O_RDWR)) < 0) {
    /* ERROR HANDLING: you can check errno to see what went wrong */
    perror("Failed to open the i2c bus");
    exit(1);
}
```

★ Initiating Communication with the AD7991

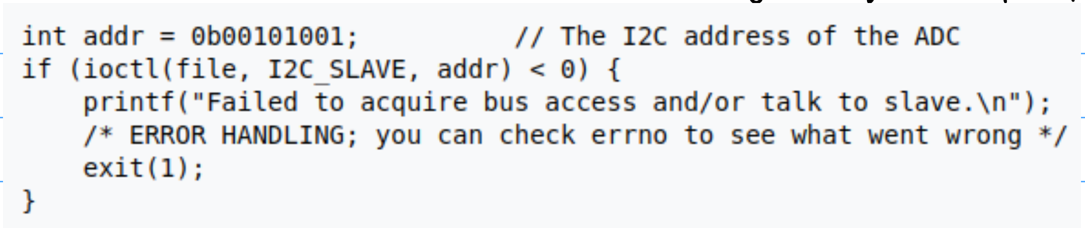
→ I2C does this by sending 7bit address followed by read/write bit.

0 → Write
1 → Read

⇒ As per data sheet, AD7991 has address of 0101001.

⇒ To use this properly zero pad the address on the left and store it as 0b00101001.

⇒ The call to read/write after IOCTL will automatically set the proper read and write bit, when signaling the peripheral.



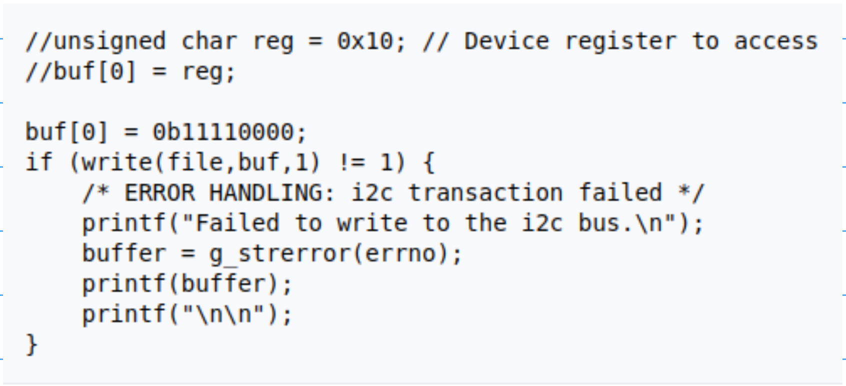
```
int addr = 0b00101001;          // The I2C address of the ADC
if (ioctl(file, I2C_SLAVE, addr) < 0) {
    printf("Failed to acquire bus access and/or talk to slave.\n");
    /* ERROR HANDLING; you can check errno to see what went wrong */
    exit(1);
}
```

★ Reading from ADC

⇒ The read system call is used to obtain data from I2C peripheral.

★ Writing to ADC

⇒ The Write system call is used to obtain data from I2C peripheral.



```
//unsigned char reg = 0x10; // Device register to access
//buf[0] = reg;

buf[0] = 0b11110000;
if (write(file,buf,1) != 1) {
    /* ERROR HANDLING: i2c transaction failed */
    printf("Failed to write to the i2c bus.\n");
    buffer = g_strerror(errno);
    printf(buffer);
    printf("\n\n");
}
```