

This is a sequence of commands used for compiling and uploading code to an AVR microcontroller, specifically the ATmega328P which is used in many Arduino boards.

1. avr-gcc -mmcu=atmega328p -Os main.c -o main.out: This command compiles your C code (main.c) into an executable file (main.out). Here’s what the options mean:  
   * -mmcu=atmega328p: This tells the compiler that the target microcontroller unit (MCU) is the ATmega328P.
   * -Os: This option tells the compiler to optimize the code for size. It tries to reduce the size of the executable while still keeping it functional.
   * -o main.out: This specifies the output file name (main.out).
2. avr-objcopy -j .text -j .data -O ihex main.out main.hex: This command converts the executable file (main.out) into a hex file (main.hex) that can be uploaded to the microcontroller. Here’s what the options mean:  
   * -j .text -j .data: These options tell avr-objcopy to only copy the .text (program code) and .data (initialized data) sections of the executable.
   * -O ihex: This specifies the output format to be Intel HEX.
3. avrdude -P /dev/ttyUSB0 -c arduino -p atmega328p -b57600 -v -D -U flash:w:main.hex:i: This command uploads the hex file to the microcontroller. Here’s what the options mean:  
   * -P /dev/ttyUSB0: This specifies the port to use for uploading.(In case it doesn’t work try using ttyACM0)
   * -c arduino: This tells avrdude that we’re using the Arduino bootloader.
   * -p atmega328p: This specifies the part number of the microcontroller.
   * -b57600: This sets the baud rate for the upload. (In case it doesn’t work try using 115200 Hz)
   * -v: This makes avrdude output verbose information, which can be useful for debugging.
   * -D: This disables auto erase for flash memory.
   * -U flash:w:main.hex:i: This tells avrdude to write to flash memory (flash:w:) the contents of main.hex (:main.hex:) in Intel Hex format (:i).