# Arduino AVR Core and platform.

# ------------------------------

# For more info:

# https://github.com/arduino/Arduino/wiki/Arduino-IDE-1.5---3rd-party-Hardware-specification

name=Marlin AVR Boards

version=1.5.6

# AVR compile variables

# ---------------------

compiler.warning\_flags=-w

compiler.warning\_flags.none=-w

compiler.warning\_flags.default=

compiler.warning\_flags.more=-Wall

compiler.warning\_flags.all=-Wall -Wextra

# Default "compiler.path" is correct, change only if you want to override the initial value

compiler.path={runtime.tools.avr-gcc.path}/bin/

compiler.c.cmd=avr-gcc

compiler.c.flags=-c -g -Os {compiler.warning\_flags} -std=gnu11 -ffunction-sections -fdata-sections -MMD -flto -fno-fat-lto-objects

compiler.c.elf.flags={compiler.warning\_flags} -Os -g -flto -fuse-linker-plugin -Wl,--gc-sections

compiler.c.elf.cmd=avr-gcc

compiler.S.flags=-c -g -x assembler-with-cpp -flto -MMD

compiler.cpp.cmd=avr-g++

compiler.cpp.flags=-c -g -Os {compiler.warning\_flags} -std=gnu++11 -fpermissive -fno-exceptions -ffunction-sections -fdata-sections -fno-threadsafe-statics -MMD -flto

compiler.ar.cmd=avr-gcc-ar

compiler.ar.flags=rcs

compiler.objcopy.cmd=avr-objcopy

compiler.objcopy.eep.flags=-O ihex -j .eeprom --set-section-flags=.eeprom=alloc,load --no-change-warnings --change-section-lma .eeprom=0

compiler.elf2hex.flags=-O ihex -R .eeprom

compiler.elf2hex.cmd=avr-objcopy

compiler.ldflags=

compiler.size.cmd=avr-size

# This can be overridden in boards.txt

build.extra\_flags=

# These can be overridden in platform.local.txt

compiler.c.extra\_flags=

compiler.c.elf.extra\_flags=

compiler.S.extra\_flags=

compiler.cpp.extra\_flags=

compiler.ar.extra\_flags=

compiler.objcopy.eep.extra\_flags=

compiler.elf2hex.extra\_flags=

# AVR compile patterns

# --------------------

## Compile c files

recipe.c.o.pattern="{compiler.path}{compiler.c.cmd}" {compiler.c.flags} -mmcu={build.mcu} -DF\_CPU={build.f\_cpu} -DARDUINO={runtime.ide.version} -DARDUINO\_{build.board} -DARDUINO\_ARCH\_{build.arch} {compiler.c.extra\_flags} {build.extra\_flags} {includes} "{source\_file}" -o "{object\_file}"

## Compile c++ files

recipe.cpp.o.pattern="{compiler.path}{compiler.cpp.cmd}" {compiler.cpp.flags} -mmcu={build.mcu} -DF\_CPU={build.f\_cpu} -DARDUINO={runtime.ide.version} -DARDUINO\_{build.board} -DARDUINO\_ARCH\_{build.arch} {compiler.cpp.extra\_flags} {build.extra\_flags} {includes} "{source\_file}" -o "{object\_file}"

## Create eeprom

recipe.objcopy.eep.pattern="{compiler.path}{compiler.objcopy.cmd}" {compiler.objcopy.eep.flags} {compiler.objcopy.eep.extra\_flags} "{build.path}/{build.project\_name}.elf" "{build.path}/{build.project\_name}.eep"

## Create hex

recipe.objcopy.hex.pattern="{compiler.path}{compiler.elf2hex.cmd}" {compiler.elf2hex.flags} {compiler.elf2hex.extra\_flags} "{build.path}/{build.project\_name}.elf" "{build.path}/{build.project\_name}.hex"

## Compute size

recipe.size.pattern="{compiler.path}{compiler.size.cmd}" -A "{build.path}/{build.project\_name}.elf"

recipe.size.regex=^(?:\.text|\.data|\.bootloader)\s+([0-9]+).\*

recipe.size.regex.data=^(?:\.data|\.bss|\.noinit)\s+([0-9]+).\*

recipe.size.regex.eeprom=^(?:\.eeprom)\s+([0-9]+).\*

## Preprocessor

preproc.includes.flags=-w -x c++ -M -MG -MP

recipe.preproc.includes="{compiler.path}{compiler.cpp.cmd}" {compiler.cpp.flags} {preproc.includes.flags} -mmcu={build.mcu} -DF\_CPU={build.f\_cpu} -DARDUINO={runtime.ide.version} -DARDUINO\_{build.board} -DARDUINO\_ARCH\_{build.arch} {compiler.cpp.extra\_flags} {build.extra\_flags} {includes} "{source\_file}"

preproc.macros.flags=-w -x c++ -E -CC

recipe.preproc.macros="{compiler.path}{compiler.cpp.cmd}" {compiler.cpp.flags} {preproc.macros.flags} -mmcu={build.mcu} -DF\_CPU={build.f\_cpu} -DARDUINO={runtime.ide.version} -DARDUINO\_{build.board} -DARDUINO\_ARCH\_{build.arch} {compiler.cpp.extra\_flags} {build.extra\_flags} {includes} "{source\_file}" -o "{preprocessed\_file\_path}"

# AVR Uploader/Programmers tools

# ------------------------------

tools.avrdude.path={runtime.tools.avrdude.path}

tools.avrdude.cmd.path={path}/bin/avrdude

tools.avrdude.config.path={path}/etc/avrdude.conf

tools.avrdude.upload.params.verbose=-v

tools.avrdude.upload.params.quiet=-q -q

tools.avrdude.upload.pattern="{cmd.path}" "-C{config.path}" {upload.verbose} -p{build.mcu} -c{upload.protocol} -P{serial.port} -b{upload.speed} -D "-Uflash:w:{build.path}/{build.project\_name}.hex:i"

tools.avrdude.program.params.verbose=-v

tools.avrdude.program.params.quiet=-q -q

tools.avrdude.program.pattern="{cmd.path}" "-C{config.path}" {program.verbose} -p{build.mcu} -c{protocol} {program.extra\_params} "-Uflash:w:{build.path}/{build.project\_name}.hex:i"

tools.avrdude.erase.params.verbose=-v

tools.avrdude.erase.params.quiet=-q -q

tools.avrdude.erase.pattern="{cmd.path}" "-C{config.path}" {erase.verbose} -p{build.mcu} -c{protocol} {program.extra\_params} -e -Ulock:w:{bootloader.unlock\_bits}:m -Uefuse:w:{bootloader.extended\_fuses}:m -Uhfuse:w:{bootloader.high\_fuses}:m -Ulfuse:w:{bootloader.low\_fuses}:m

tools.avrdude.bootloader.params.verbose=-v

tools.avrdude.bootloader.params.quiet=-q -q

tools.avrdude.bootloader.pattern="{cmd.path}" "-C{config.path}" {bootloader.verbose} -p{build.mcu} -c{protocol} {program.extra\_params} "-Uflash:w:{runtime.platform.path}/bootloaders/{bootloader.file}:i" -Ulock:w:{bootloader.lock\_bits}:m

tools.hidloader.cmd.path=/usr/local/bin/HIDUploader

tools.hidloader.upload.params.verbose=-v

tools.hidloader.upload.params.quiet=

tools.hidloader.upload.pattern="{cmd.path}" --upload -mmcu={build.mcu} {upload.verbose} -w "{build.path}/{build.project\_name}.hex"

tools.nativehid.program.params.verbose=-v

tools.nativehid.program.params.quiet=-q -q

tools.hidloader.program.params.verbose=-v

tools.hidloader.program.params.quiet=-q -q

tools.hidloader.program.pattern="{cmd.path}" -mmcu={build.mcu} {upload.verbose} -w "{build.path}/{build.project\_name}.hex"

tools.hidloader.erase.params.verbose=-v

tools.hidloader.erase.params.quiet=-q -q

tools.hidloader.erase.pattern="{cmd.path}" --erase "-C{config.path}" {erase.verbose} -p{build.mcu} -c{protocol} {program.extra\_params} -e -Ulock:w:{bootloader.unlock\_bits}:m -Uefuse:w:{bootloader.extended\_fuses}:m -Uhfuse:w:{bootloader.high\_fuses}:m -Ulfuse:w:{bootloader.low\_fuses}:m

# USB Default Flags

# Default blank usb manufacturer will be filled it at compile time

# - from numeric vendor ID, set to Unknown otherwise

build.usb\_manufacturer=

build.usb\_flags=-DUSB\_VID={build.vid} -DUSB\_PID={build.pid} '-DUSB\_MANUFACTURER={build.usb\_manufacturer}' '-DUSB\_PRODUCT={build.usb\_product}'