

Post-Generation Scripts

Introduction

We use one post code generation script. We do this because we do not want to write code inside the auto-generated code, and this helps with that. If you are on Linux or (possibly, untested but likely) mac, you can refer to the script in the cubeMX software by going into Project Manager -> Code Generator -> User Actions -> After Code Generation. If you are not on a UNIX based system you have to run it every time after generating code manually from the folder where the file is located. The file is located under scripts and is called post_code_generation.bash. For mac, the script is called post_code_generator_mac.bash, because there are some small formatting changes, more explained in Mac Changes.

Functions

The script has 4 different functions

Renaming main files

It starts by renaming all main.h and main.c files that are in components. It also saved all boards that have main.c files, because those are the newly generated boards. This way you don't do certain actions on the files twice, if you would run the script again.

```
while IFS= read -r FILE; do
    # Extract the basename (filename without path)
    base="$(basename "$FILE")"

    if [[ "$base" == "main.c" ]]; then

        subdir="${FILE#"${BASE}"/}" # Path from the board dir
        main_dir="${subdir%/*}" # The board dir
        GENERATED_BOARDS+=("$main_dir") # Gets all boards that are generated again, and thus have
a main

        dir=$(dirname "$FILE") # Get directory of the file
        mv "$FILE" "$dir/cubemx_main.c"
        echo "Renamed $FILE to $dir/cubemx_main.c"
    fi
done
```

```

if [[ "$base" == "main.h" ]]; then
    dir=$(dirname "$FILE") # Get directory of the file
    mv "$FILE" "$dir/cubemx_main.h"
    echo "Renamed $FILE to $dir/cubemx_main.h"
fi
done < <(find "$BASE" -type f)

```

Adding firmware definitions

Certain constants might have to be set in the main.h files from cubemx. This happens most likely because of the order in which the files are build, but I am not totally sure. However, if you do need to have some constants set in the main.h file, you can add them to any file in the folder called "firmware_definitions" in the common folder of components. This second code block adds it to the .h file.

```

find "$BASE" -type d -name firmware_definitions | while read -r FW_DIR; do
    BOARD_DIR_PATH="$(dirname "$FW_DIR")"
    BOARD_DIR="${BOARD_DIR_PATH#"${BASE}"/}"
    if printf '%s\n' "$COMMON_COMPONENT" "${GENERATED_BOARDS[@]}" | grep -Fx "$BOARD_DIR" >
/dev/null; then
        BOARD_DIR_PATHS=("$BOARD_DIR_PATH")

        if [[ "$BOARD_DIR" == "$COMMON_COMPONENT" ]]; then
            BOARD_DIR_PATHS=("${GENERATED_BOARDS[@]}/#/"${BASE}/")
        fi
        for BOARD_DIR_PATH in "${BOARD_DIR_PATHS[@]}; do

            CUBEMX_FILE="$BOARD_DIR_PATH/firmware/Core/Inc/cubemx_main.h"

            # Skip if cubemx file does not exist
            [[ -f "$CUBEMX_FILE" ]] || continue

            echo "Appending firmware_definitions to: $CUBEMX_FILE"

            TMP_FILE="$(mktemp)"

            head -n -1 "$CUBEMX_FILE" >> "$TMP_FILE"

            echo -e "\n/* ---- START firmware_definitions ---- */\n" >> "$TMP_FILE"
            find "$FW_DIR" -type f -exec cat {} \; >> "$TMP_FILE"

```

```

echo -e "\n/* ---- END firmware_definitions ---- */\n" >> "$TMP_FILE"

tail -n -1 "$CUBEMX_FILE" >> "$TMP_FILE"


# 3) Replace original file
mv "$TMP_FILE" "$CUBEMX_FILE"
done
fi
done

```

Adding static wrappers

Some functions generated by cubemx you do need, but they are static so you cannot use them outside of the main file. To still be able to do that, the script adds wrappers for those files.

```

find "$BASE" -type f -name "cubemx_main.c" | while read -r FILE; do
    subdir="${FILE#"${BASE}"/}" # Path from the board dir
    BOARD_DIR="${subdir%/*}" # The board dir
    if printf '%s\n' "${GENERATED_BOARDS[@]}" | grep -Fx "$BOARD_DIR" > /dev/null; then
        TMP_FILE="$(mktemp)"
        echo "READING $FILE"
        while read -r line; do
            echo "$line" >> "$TMP_FILE"
            if [[ "$line" =~ ^[[:space:]]*static[[:space:]]+[a-zA-Z][a-zA-Z0-9]*[[:space:]]+[a-zA-Z][a-zA-Z0-9]*\([^\\]*\\);[[:space:]]*$ ]]; then
                # Remove 'static' and trailing ';'

                echo "Static function found: $line"
                proto=$(echo "$line" | sed -E 's/^[[:space:]]*static[[:space:]]+//;
s/[[:space:]]*$//')

                # Extract function name
                name=$(echo "$proto" | sed -E 's/.*[[:space:]]+([a-zA-Z][a-zA-Z0-9]*)\(.*/\1/')

                # Extract return type
                ret=$(echo "$proto" | sed -E "s/[[:space:]]+$name\(.*/")

                # Extract argument list
                args=$(echo "$proto" | sed -E "s/.*$name\((.*)\)/\1/")
            fi
        done
    fi
done

```

```

# Build argument names (remove types)
call_args=$(echo "$args" | sed -E 's/[a-zA-Z_][a-zA-Z0-9_]*[[:space:]]+//g')
if [[ "$call_args" == "void" ]]; then
    call_args=""
fi

echo "$ret ${name}_wrapper($args) {" >> "$TMP_FILE"
if [[ "$ret" == "void" ]]; then
    echo "    $name($call_args);" >> "$TMP_FILE"
else
    echo "    return $name($call_args);" >> "$TMP_FILE"
fi
echo "}" >> "$TMP_FILE"
echo >> "$TMP_FILE"
echo "added wrapper for static function ${name} in ${FILE}"
fi
done < "$FILE"
mv "$TMP_FILE" "$FILE"
fi
done

```

Changing the includes

Because of the name change from main.c/h, to cubemx_main.c/h, the includes are now wrong. This last code block changes all the includes to the right name.

```

while IFS= read -r FILE; do
    subdir="${FILE#"${BASE}"/}" # Path from the board dir
    BOARD_DIR="${subdir%/*}" # The board dir
    if printf '%s\n' "${GENERATED_BOARDS[@]}" | grep -Fx "$BOARD_DIR" > /dev/null; then
        sed -i 's/#include "main.h"/#include "cubemx_main.h"/g' "$FILE"
        echo "Updated include in $FILE"
    fi
done < <(grep -rl '#include "main.h"' ../components/)

```

MAC Changes

1) Added `#!/usr/bin/env bash` in the first line

2) changed `sed -i 's/#include "main.h"/#include "cubemx_main.h"/g' "$FILE"`

to `sed -i '' 's/#include "main.h"/#include "cubemx_main.h"/g' "$FILE"` because mac uses a different version of sed.

Revision #6

Created 2026-04-16 08:03:56 UTC by Nikolaos Diamantopoulos

Updated 2026-04-28 10:49:26 UTC by Irem Civginer