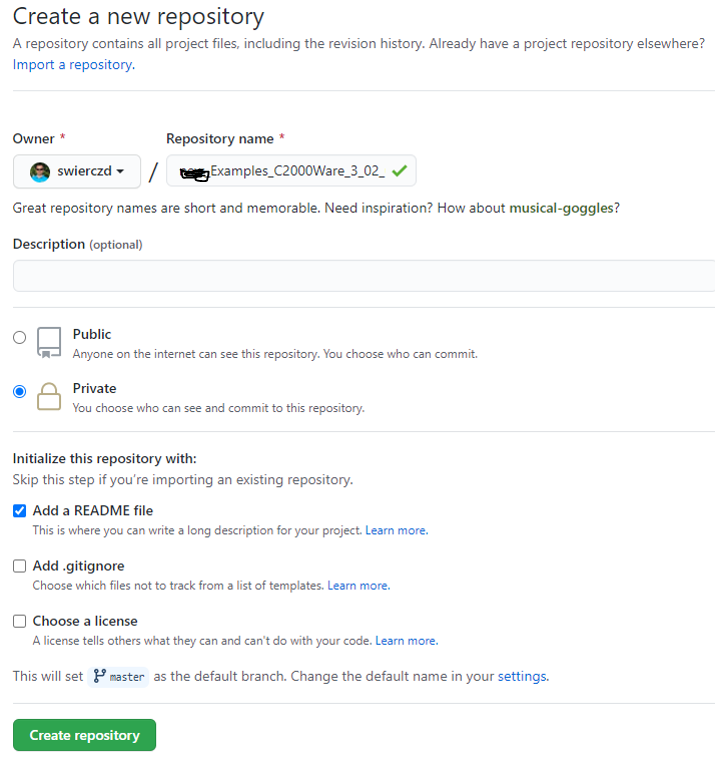
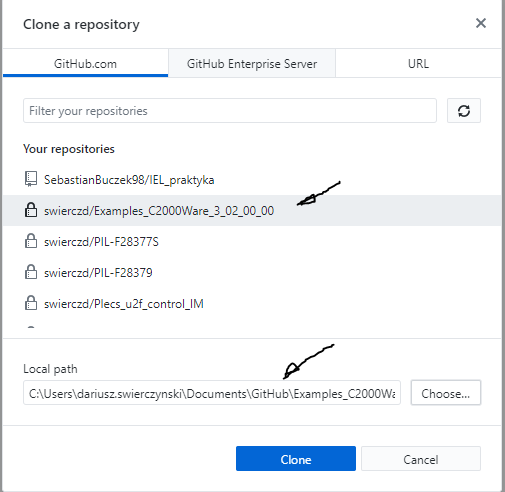
Tworzę nowe repozytorium na GitHub

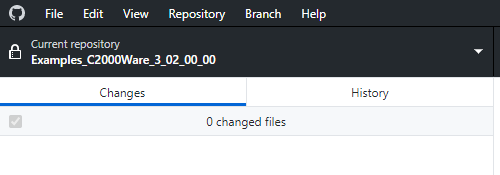


Plik README.md ---tworzymy w nim dłuższy opis projektu

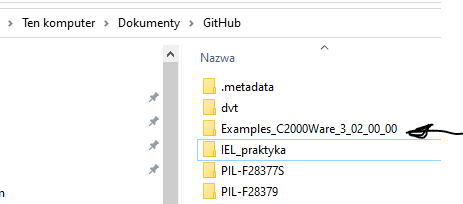
Po utworzeniu uruchamiam zainstalowanego na komputerze GitHub-a. Wybieram File/Clone Repository. Wybieramy z listy właściwe Repository. Podajemy lokalną ścieżkę gdzie znajdować się będzie clone.

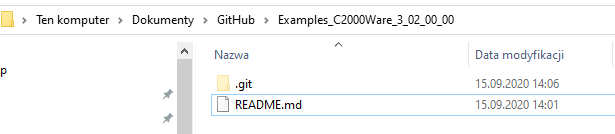


Po sklonowaniu w oprogramowaniu GitHub zobaczymy następujące okno



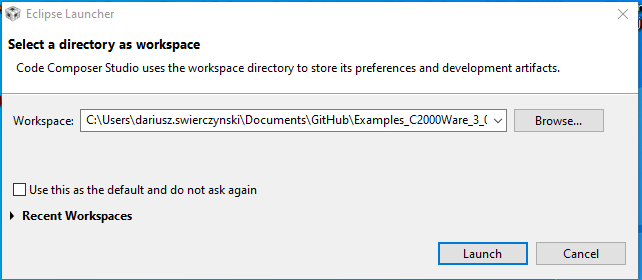
W katalogu GitHub (miejsce na clone-y) utworzony został katalog



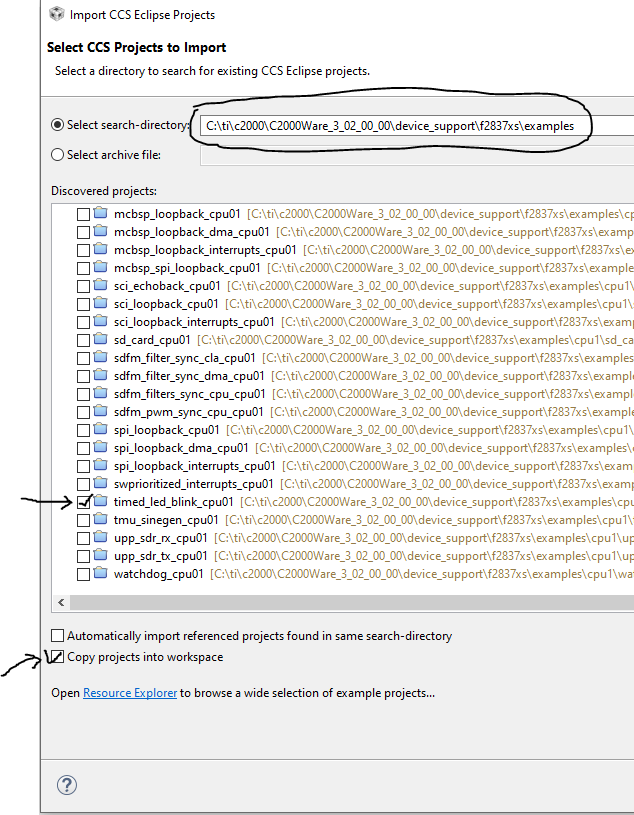


Plik README.md ---tworzymy w nim dłuższy opis projektu

Uruchamiamy CCS v8.3.1 . Wybieramy katalog workspace. W naszym przypadku to

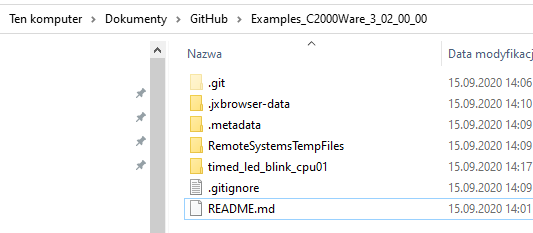
C:\Users\dariusz.swierczynski\Documents\GitHub\Examples\_C2000Ware\_3\_02\_00\_00

Teraz należy zaimportować przykładowy projekt z C2000Ware dla procesora z F2

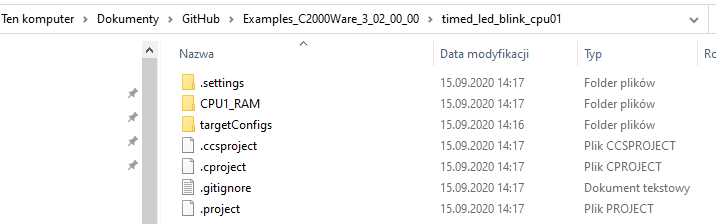


Zaznaczenie opcji kopiuj oznacza, że wszystkie pliki źródłowe zaimportowane zostano do workspace jako kopie a nie jako linki do źródłowego projektu z C2000Wave. W naszym przypadku workspace to C:\Users\dariusz.swierczynski\Documents\GitHub\Examples\_C2000Ware\_3\_02\_00\_00. Katalog GtHub to clon repozytorium na GitHub

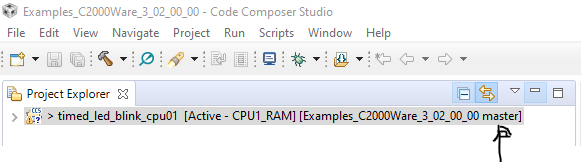
W katalogu utworzyły się dodatkowa struktura . Projekt timed\_led\_blink\_cpu01 został zaimportowany do workspace.



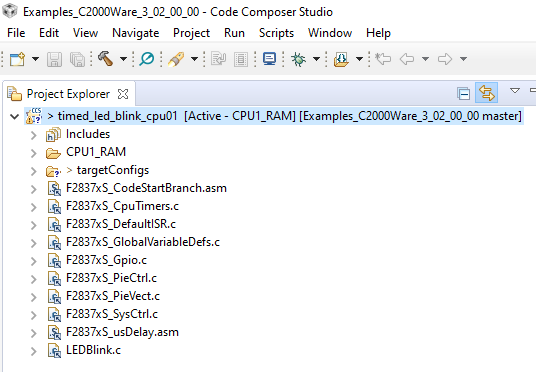
Katalog projektu ma następującą strukturę

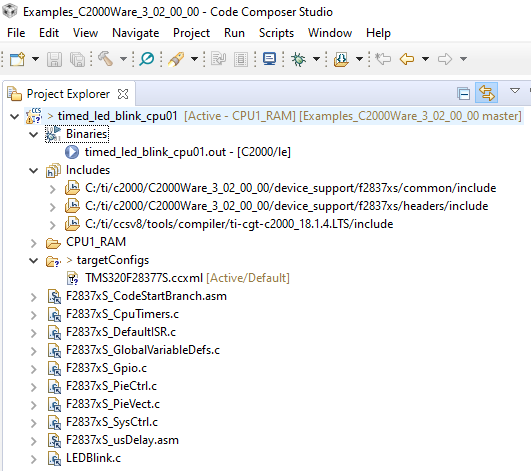


Po zaimportowaniu projektu w CCS v6.2 wygląd to następująco

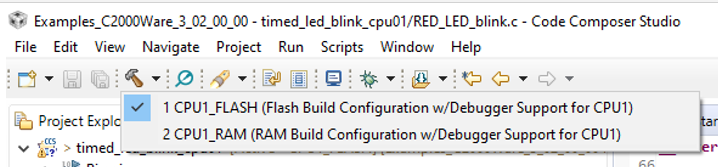


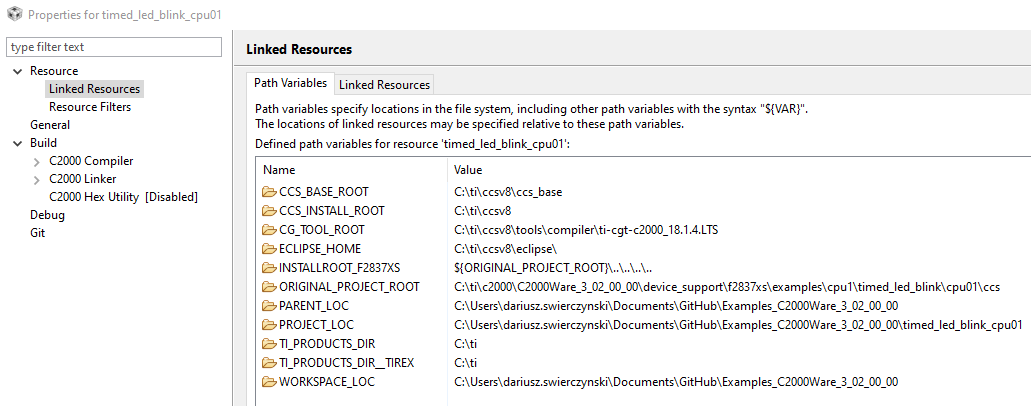
W nawiasach kwadratowych [……master] oznacza, ze mamy w tym katalogu workspace katalog time\_led\_blink\_cpu01, który jest clonem repozytorium na GitHub. MASTER

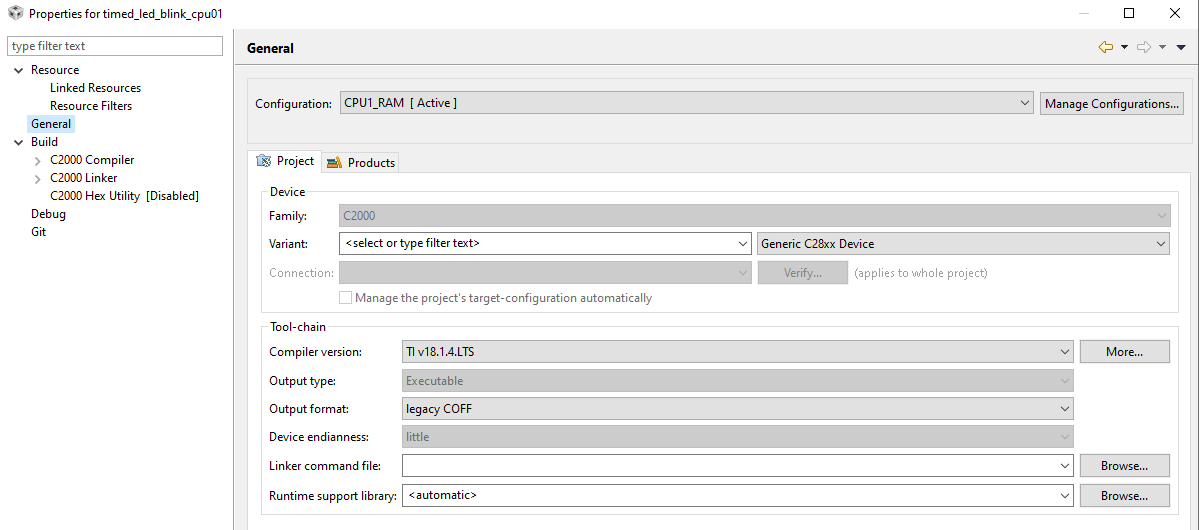


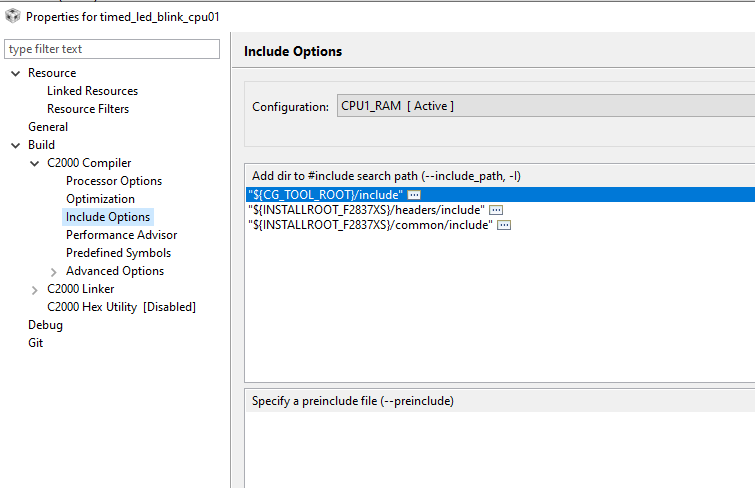


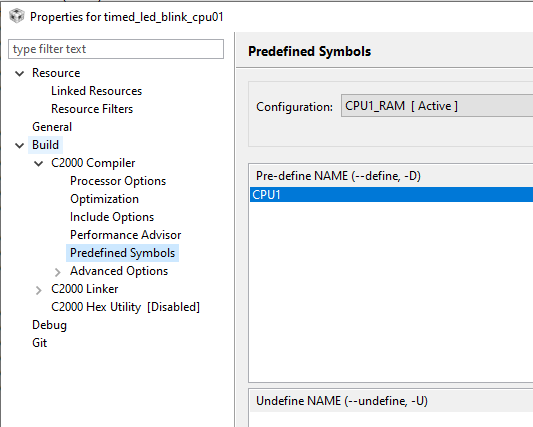
Ustawienia konfiguracyjne CPU1\_RAM

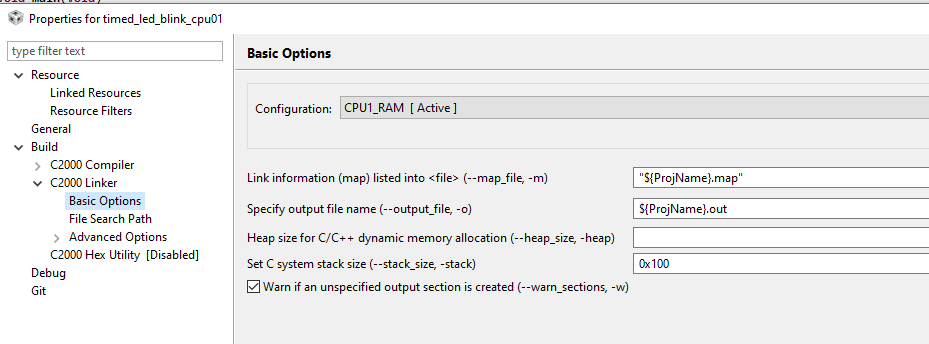


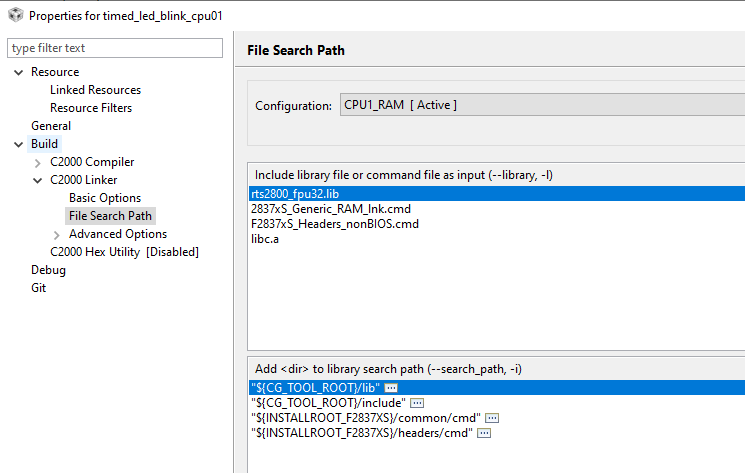


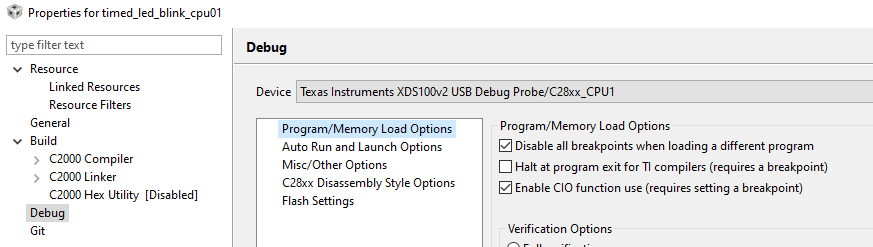


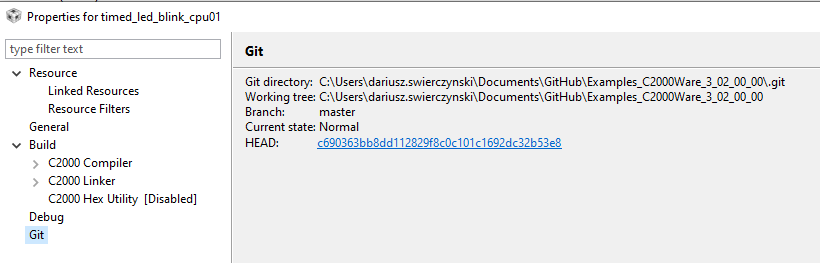




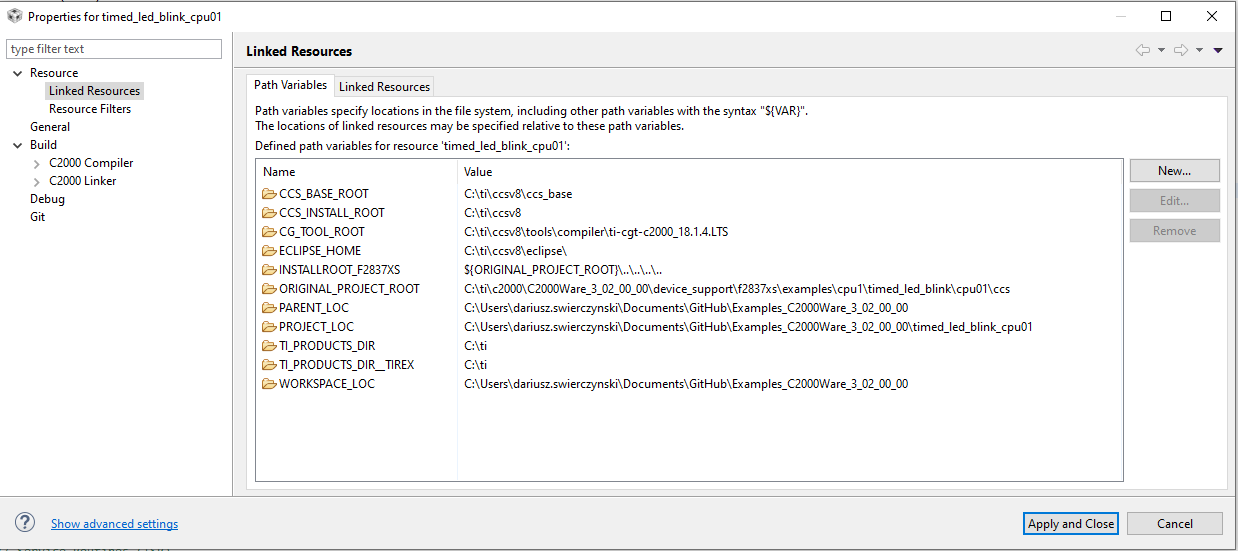


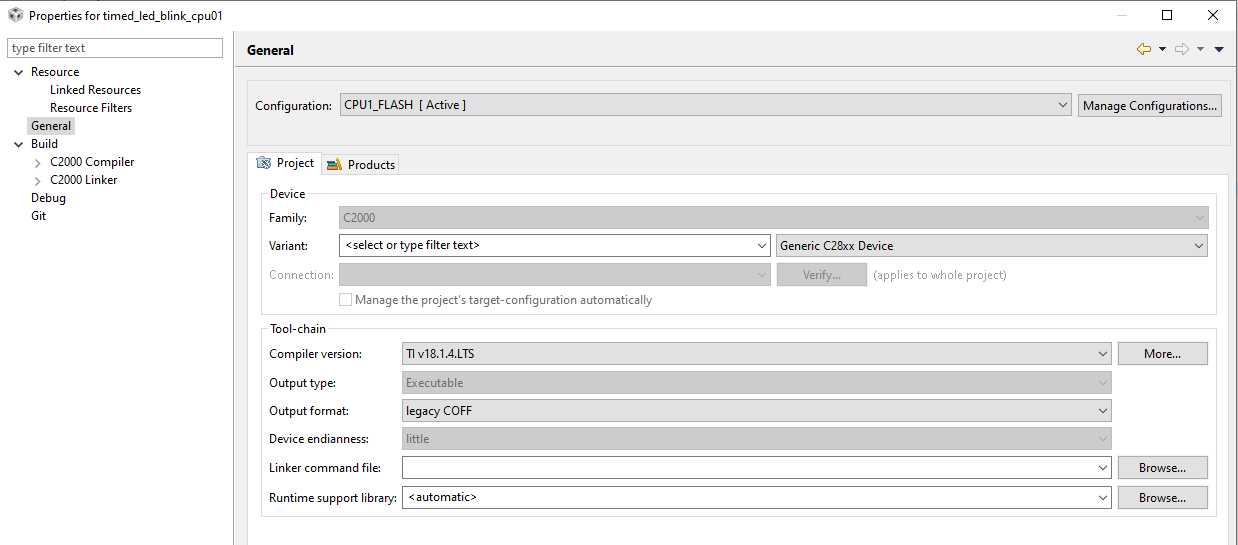


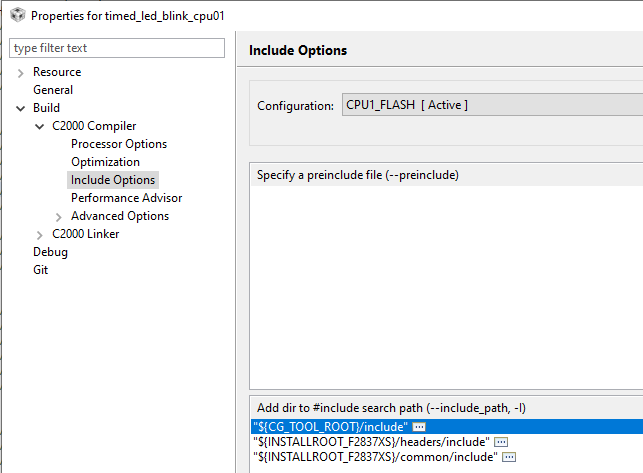


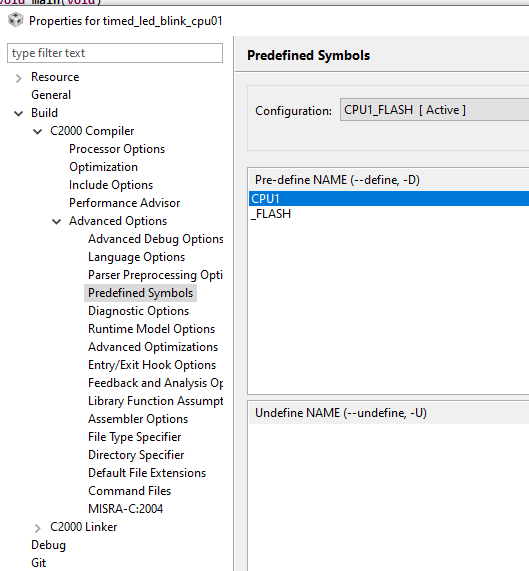


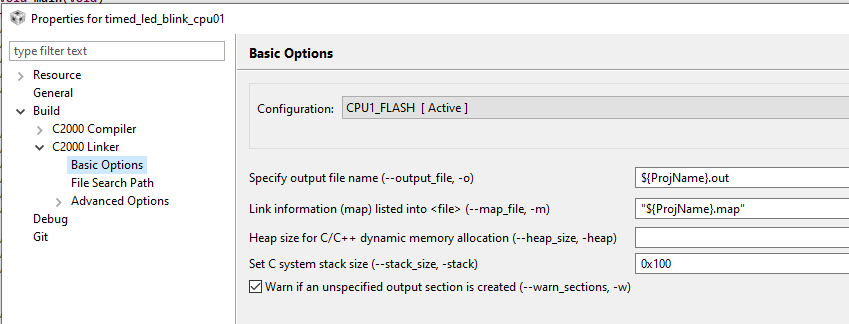
Ustawienia konfiguracyjne CPU1\_FLASH

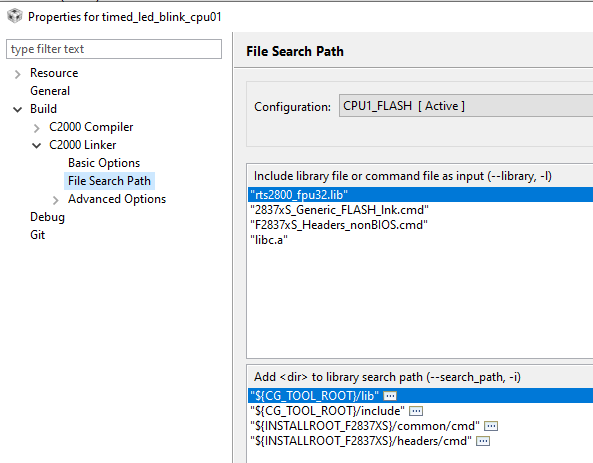


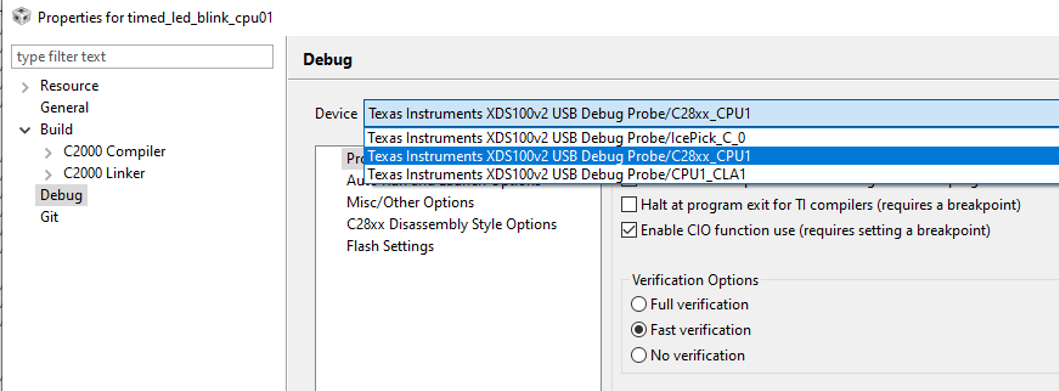


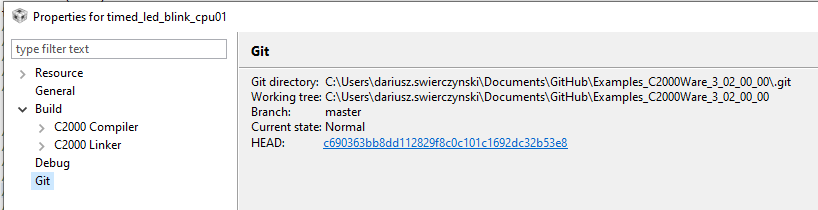




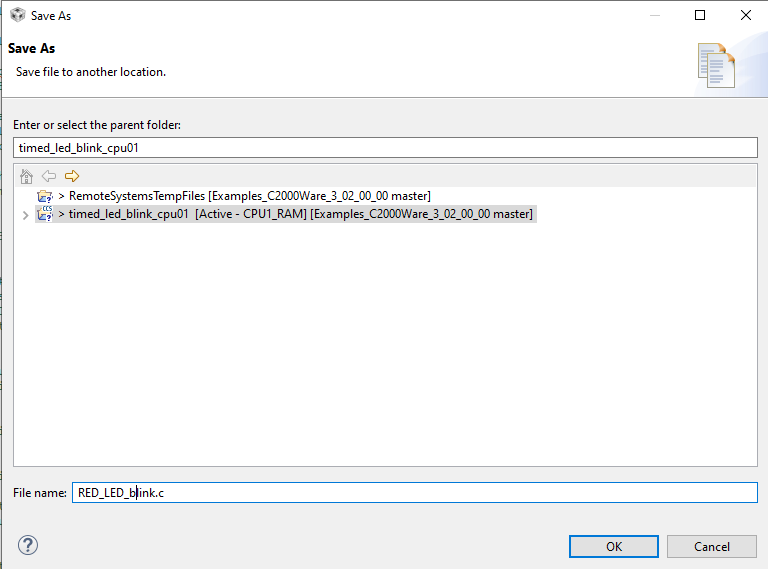




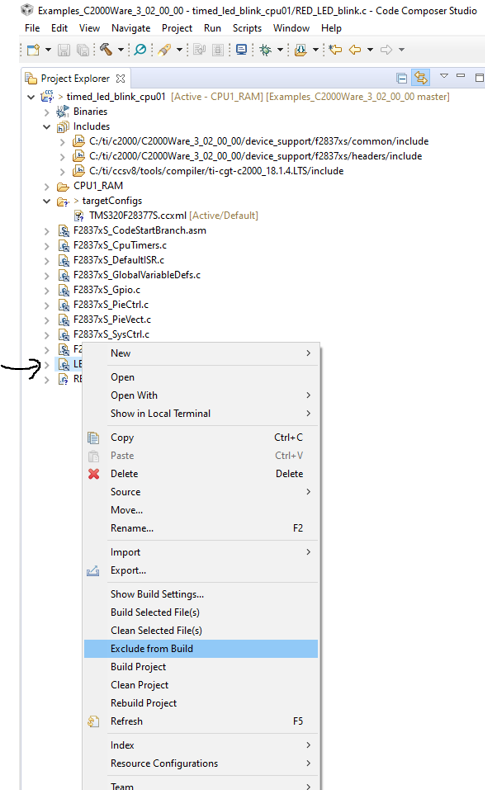




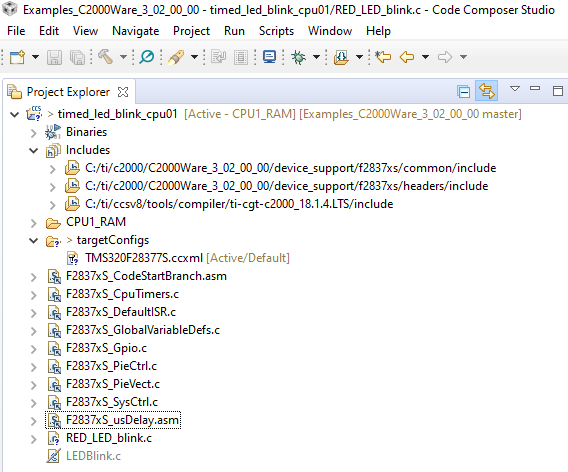
Główny program to LEDBlink.c Otwórzmy go i zapiszmy go pod inną nazwą RED\_LED\_blink



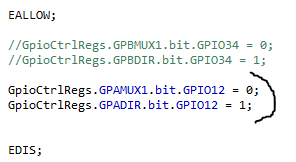
Teraz trzeba plik LEDBlink.c wyłączyć z projektu.



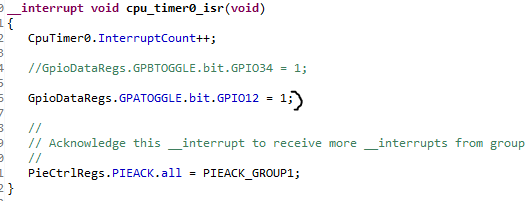
Teraz będzie szary



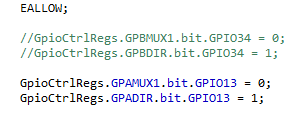
Dokonamy teraz edycji pliku RED\_LED\_blink.c

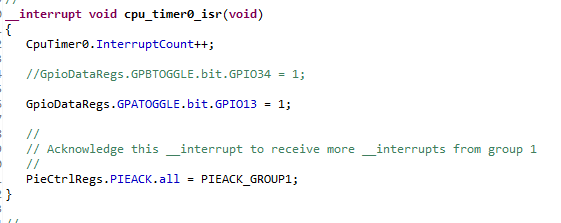


Dla diody RED



Dla diody BLUE





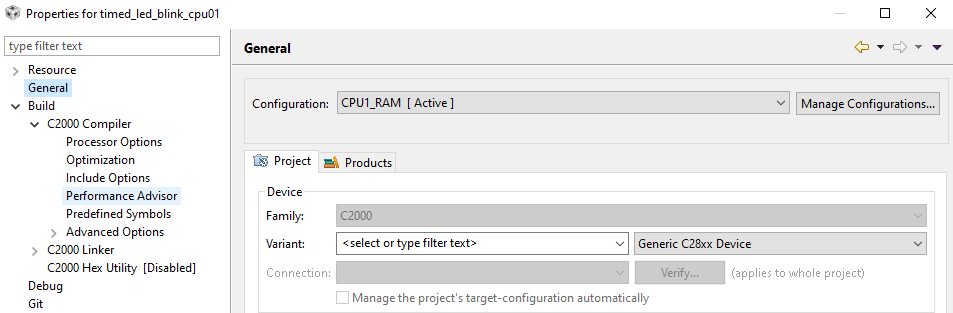
Dla płytki LaunchPadXL 28377S na GPIO 12 i 13 podłączone są diody. Na GPIO 12 jest to dioda czerwona (RED\_LED) na GPIO 13 jest dioda niebieska (BLUE\_LED).

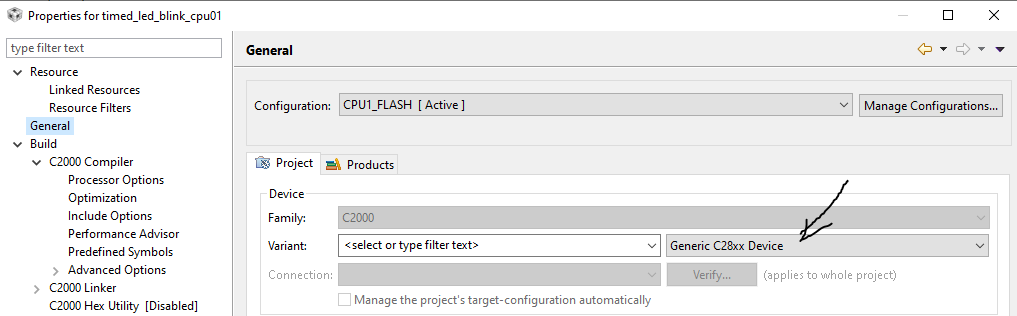
W wyniku kompilacji projektu i wygenerowaniu pliku binarnego out

Następnie należy zaprogramować procesor do CPU1\_RAM-u mikroprocesora. Po uruchomieniu Odpowiednia dioda zapala się i gaśnie z zdefiniowaną w programie częstotliwością

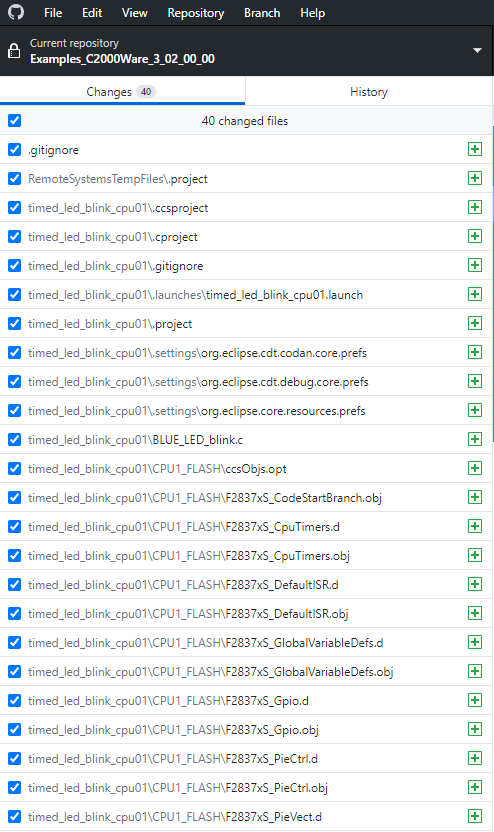
Można również zaprogramować procesor w konfiguracji CPU1\_FLASH. Pamiętać należy, ze w katalogu projektu może występować tylko jeden plik C w którym jest funkcja main. Pozostałe należy wyłączyć z projektu.Zaprogramowanie procesora w konfiguracji CPU1\_FLASH oznacza, ze program jest wpisany na stałe (po zresetowaniu urządzenia LaunchPadXL program zaczyna działać od razu)

Należy zwrócić uwagę na to , że generacja dla Variant -----Generic C28xx Device

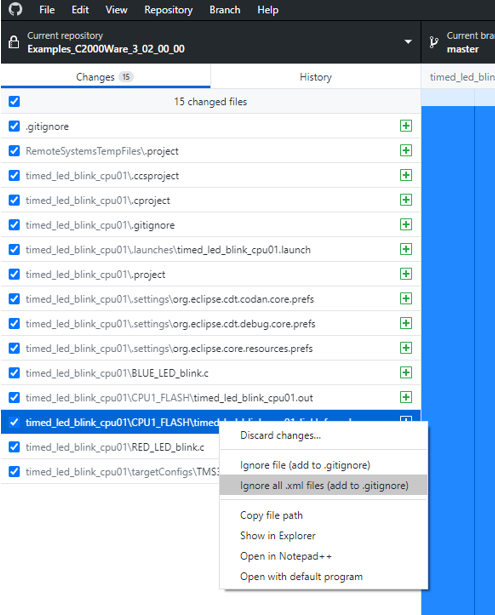




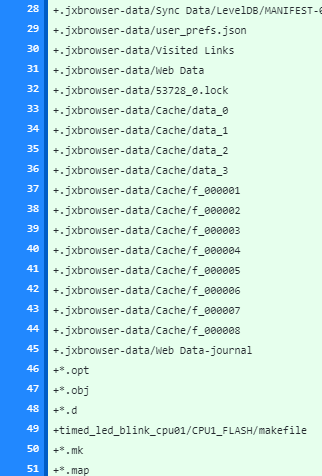
Po dokonaniu kompilacji w katalogi workspace, który jest jednocześnie klonem repozytorium na GitHub (Example\_C2000Ware\_3\_02\_00\_00) zobaczym, ze GitHub zobaczył zmiany w katalogu.(Changes)



Należy teraz ustawić, które typy plików będą wersjonowane

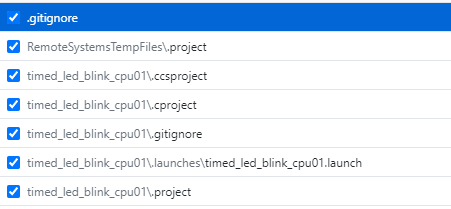


W piku .gitignore można zobaczyć jakie są ustawienia co typu pliku, które są ignorowane.

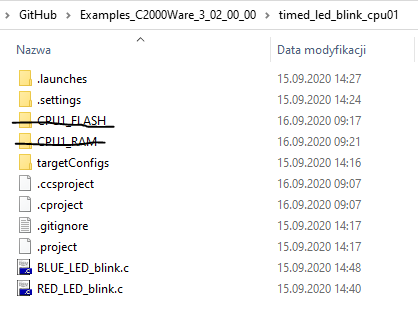


Z naszego punktu widzenia musimy wersjonować pliki typu C oraz H

Nie wiem jeszcze jak jest z plikami ustawieniowymi dla projektu

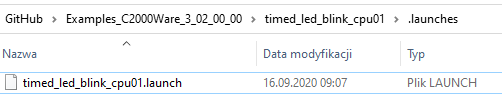


W katalogu projektowy powstała struktura

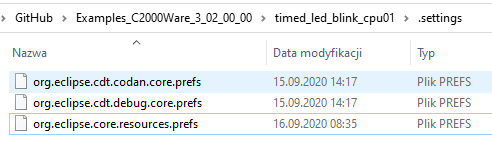


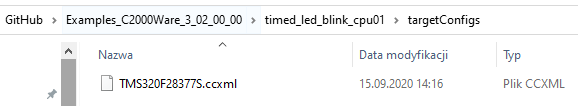
Na pewno nie musimy do GitHub wysyłać katalogów CPU1\_RAM oraz CPU1\_FLASH

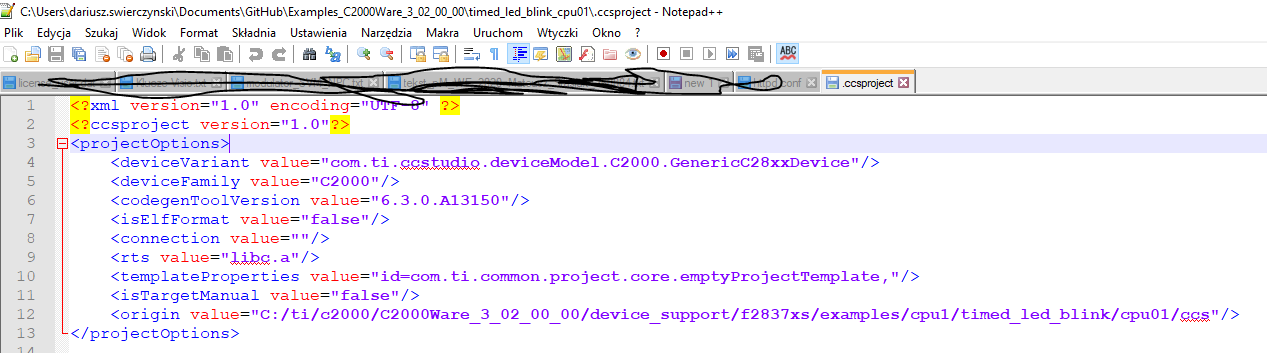
Katalog .launches zawiera



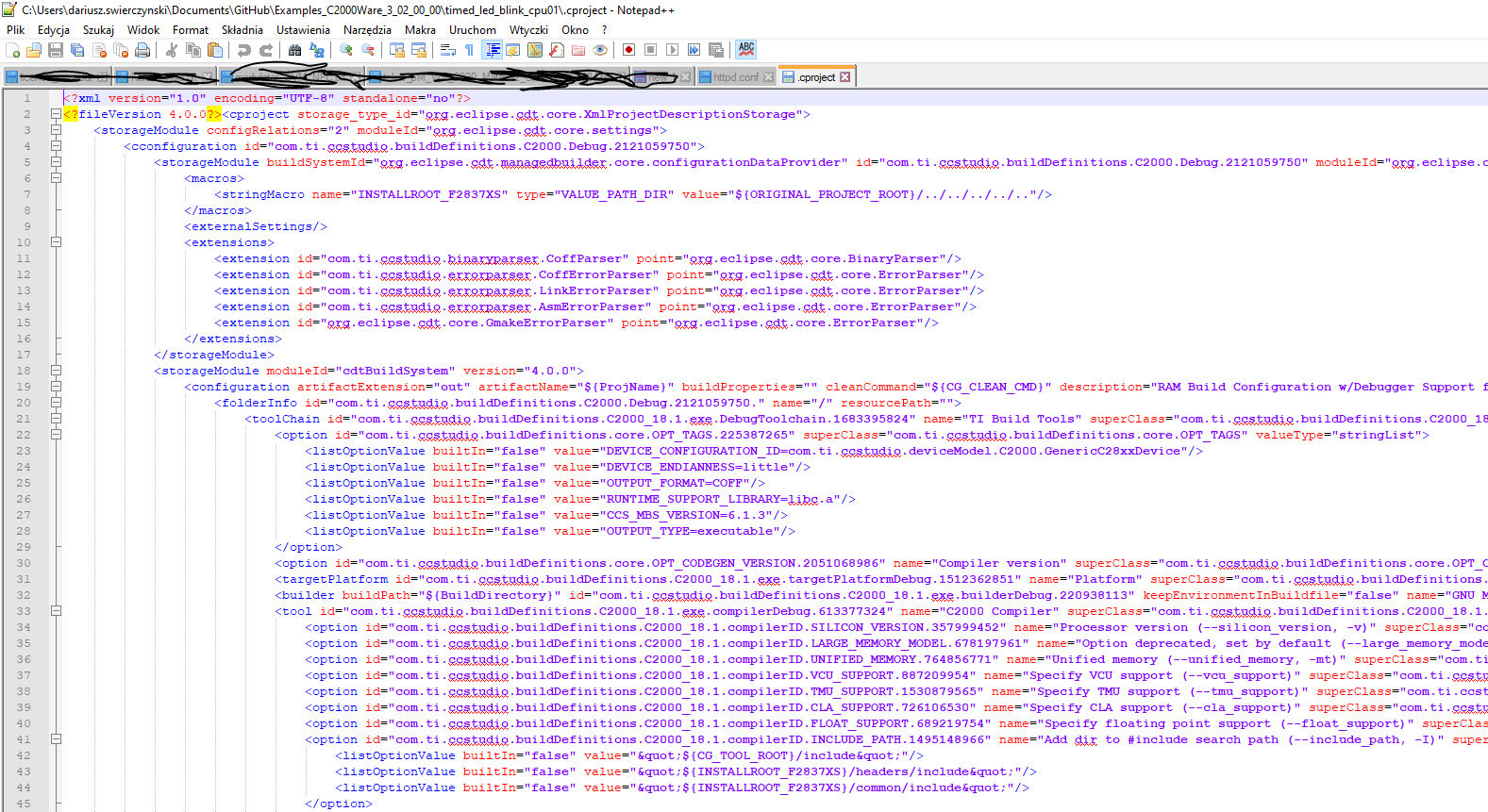
Po dokonaniu edycji tego pliku można zobaczyć konfigurację dotyczącą lunchpad



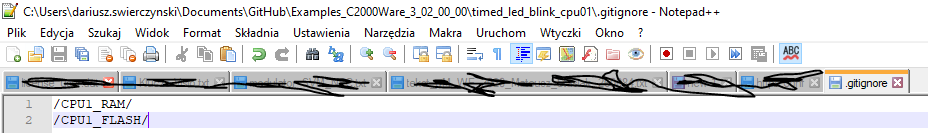




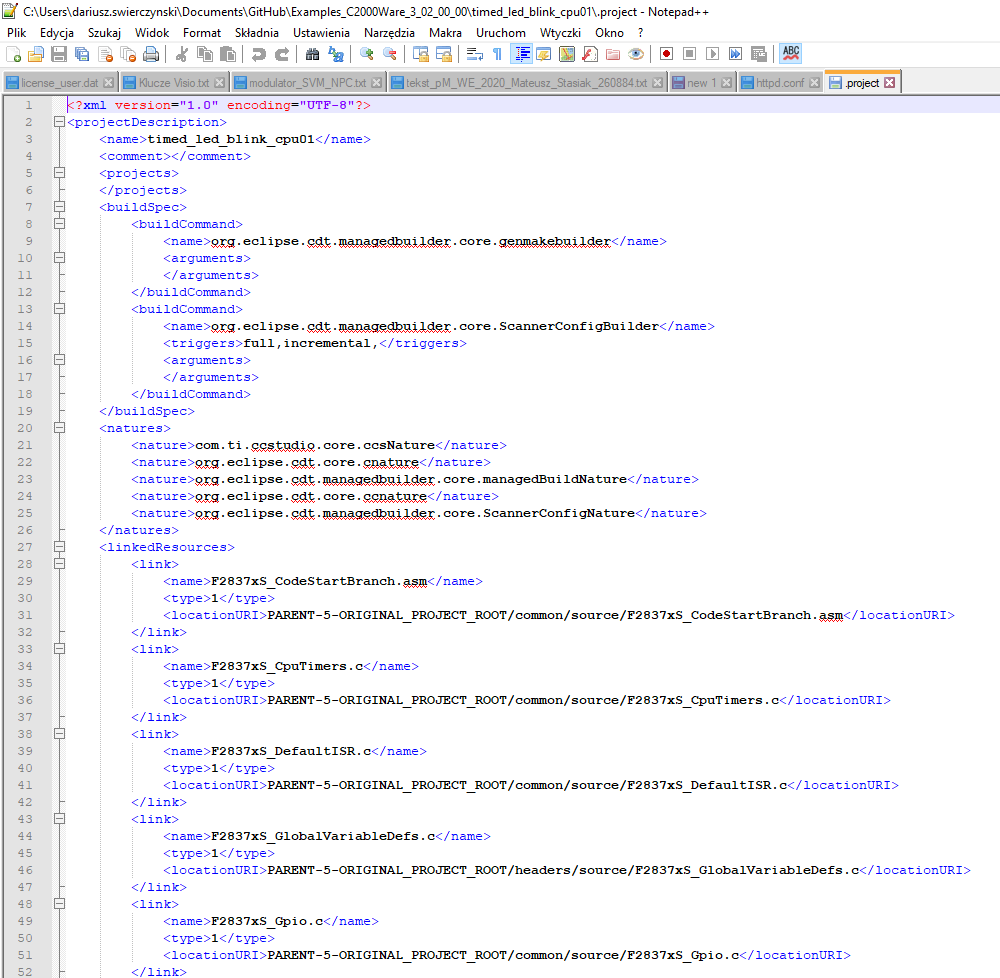
W tym pliku są ustawienia konfiguracyjne co do Buildowania



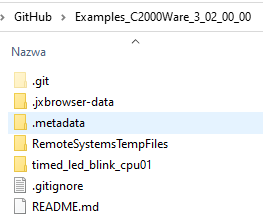
W tym pliku można stawić jakie katalogi będą ignorowane. Dotyczy to katalogu gdzie znajduje się projekt



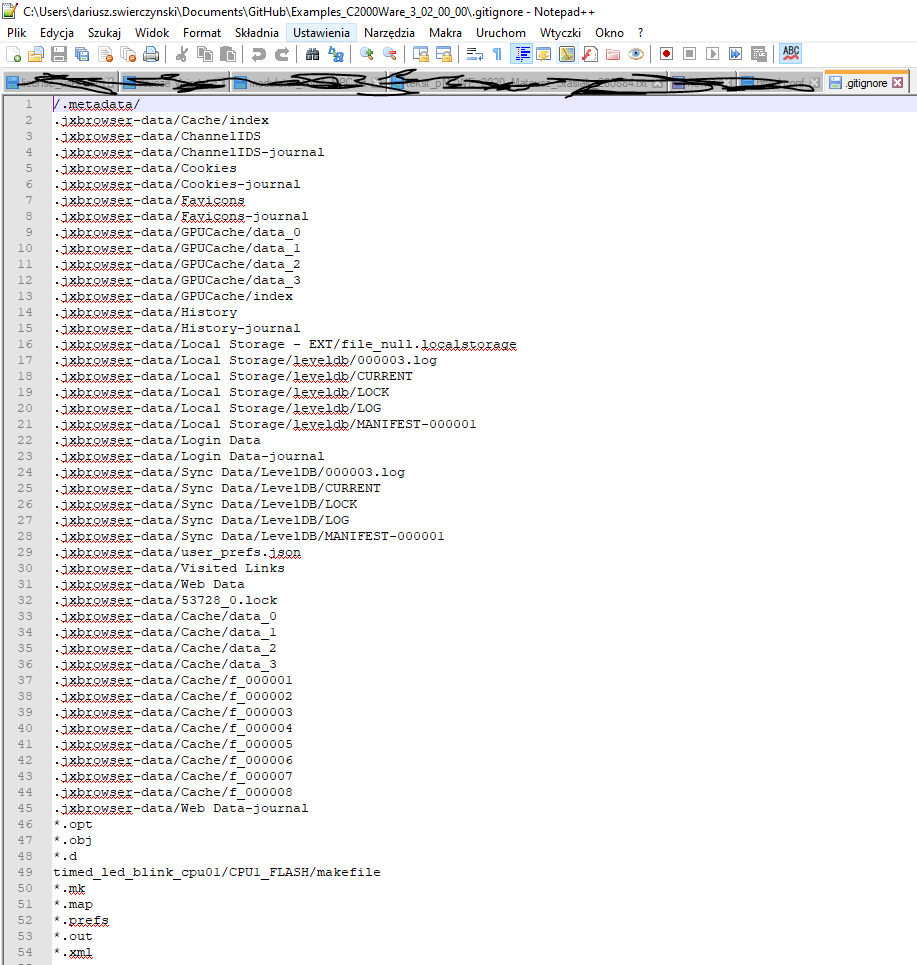
W tym pliku mam



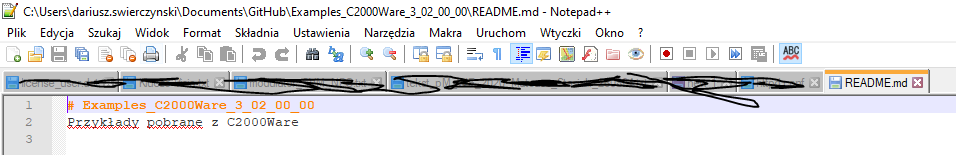
W katalogu workspace mamy następującą strukturę



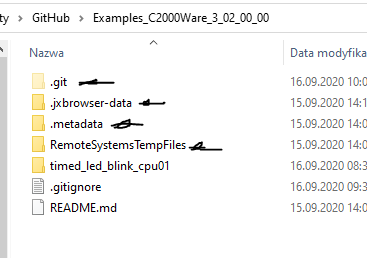
Na uwagę zasługuje plik ,gitignore. Tutj są wszystkie ignorowania plików lub katalogów co do workspace



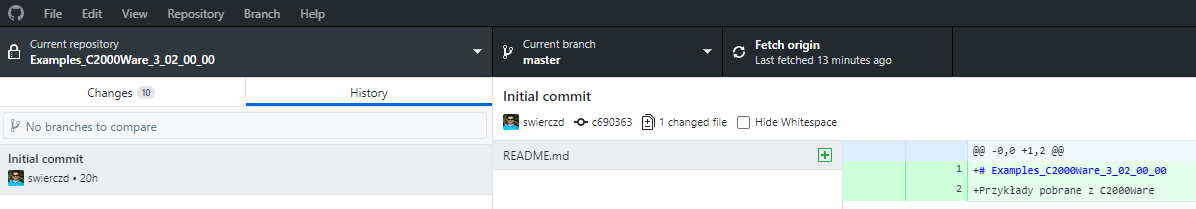
Plik READ.md zawiera opis przestrzni workspace



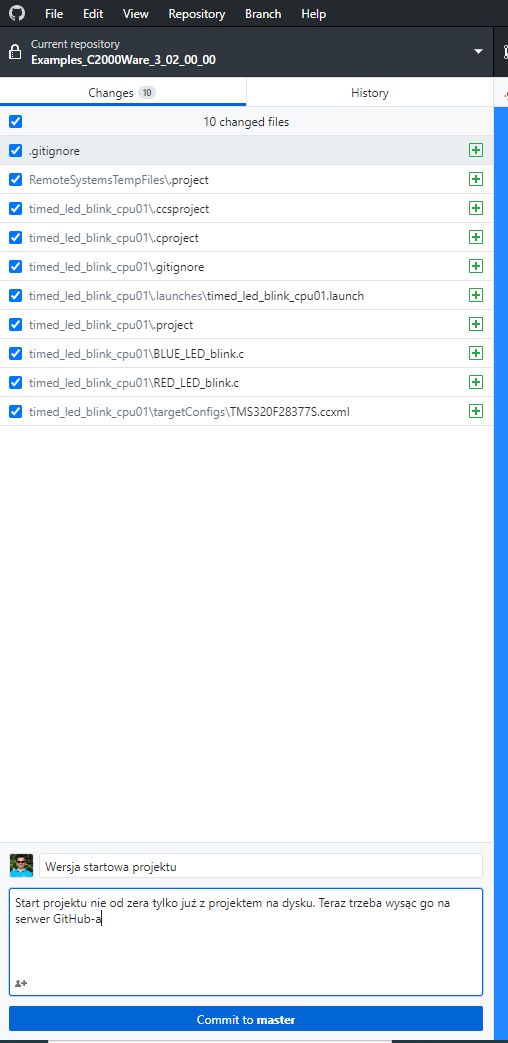
Pozostałe pliki to konfiguracyjne dla workspace oraz dla git



W aktualnej historii repozytorium mamy

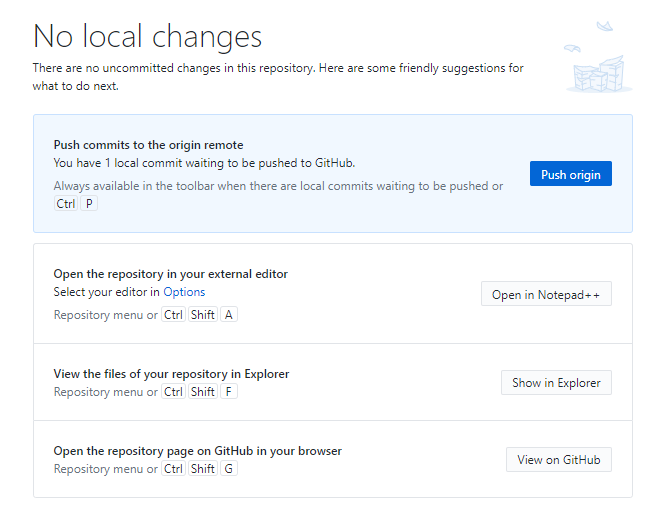


Dalej trzeba zrobić comitt-a.



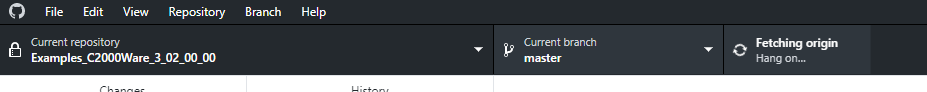
Klikamy Commit to master

Pojawia się okno



Klikamy Push origin (Pchaj oryginalna wersję na serwer)

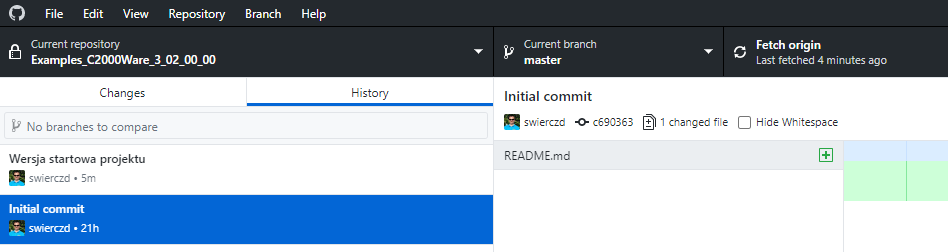
W oknie widzimy, że następuje proces Fatching origin (Pobieram źródło)

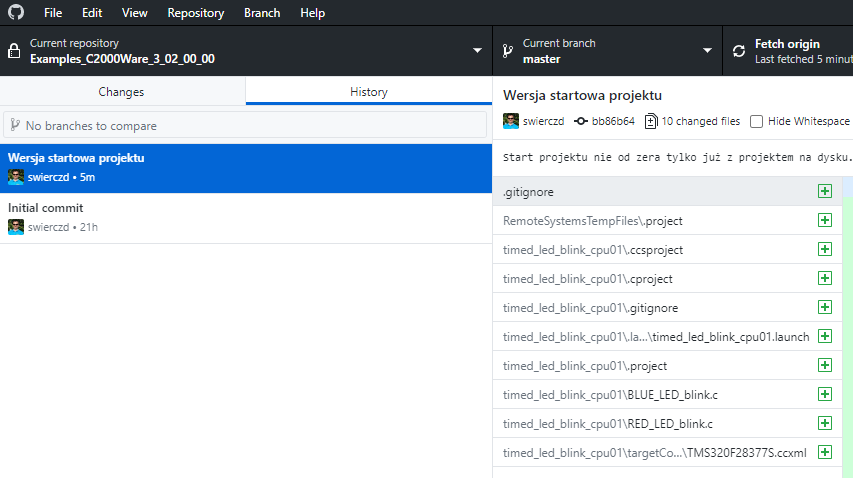


Po zakończeniu procesu

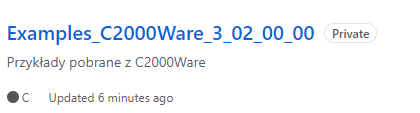


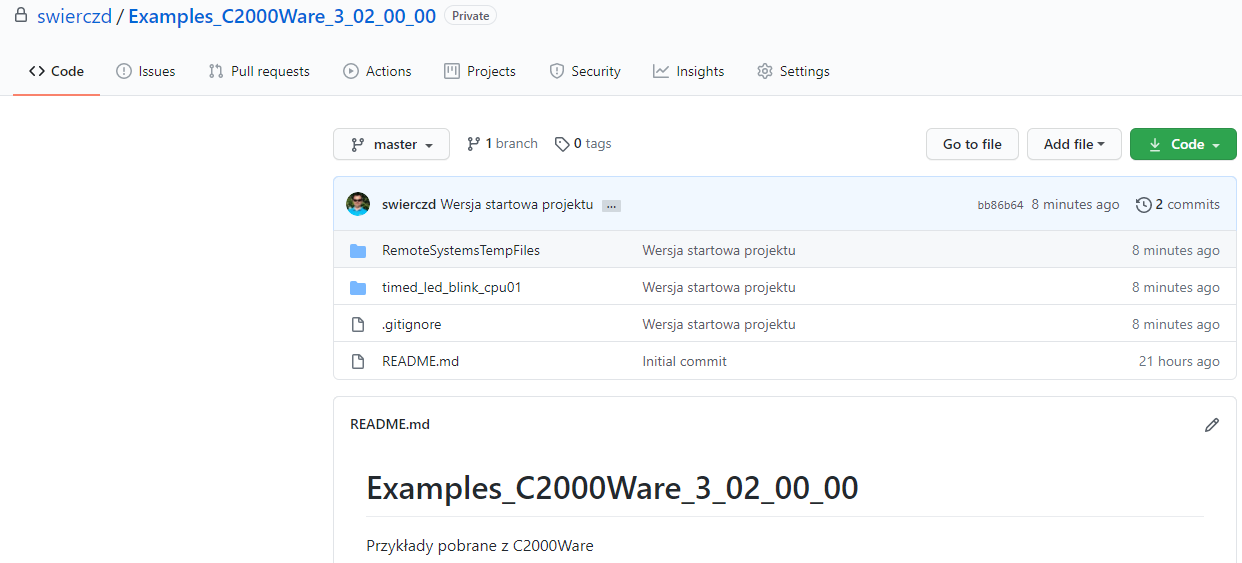
W zakładce History mamy

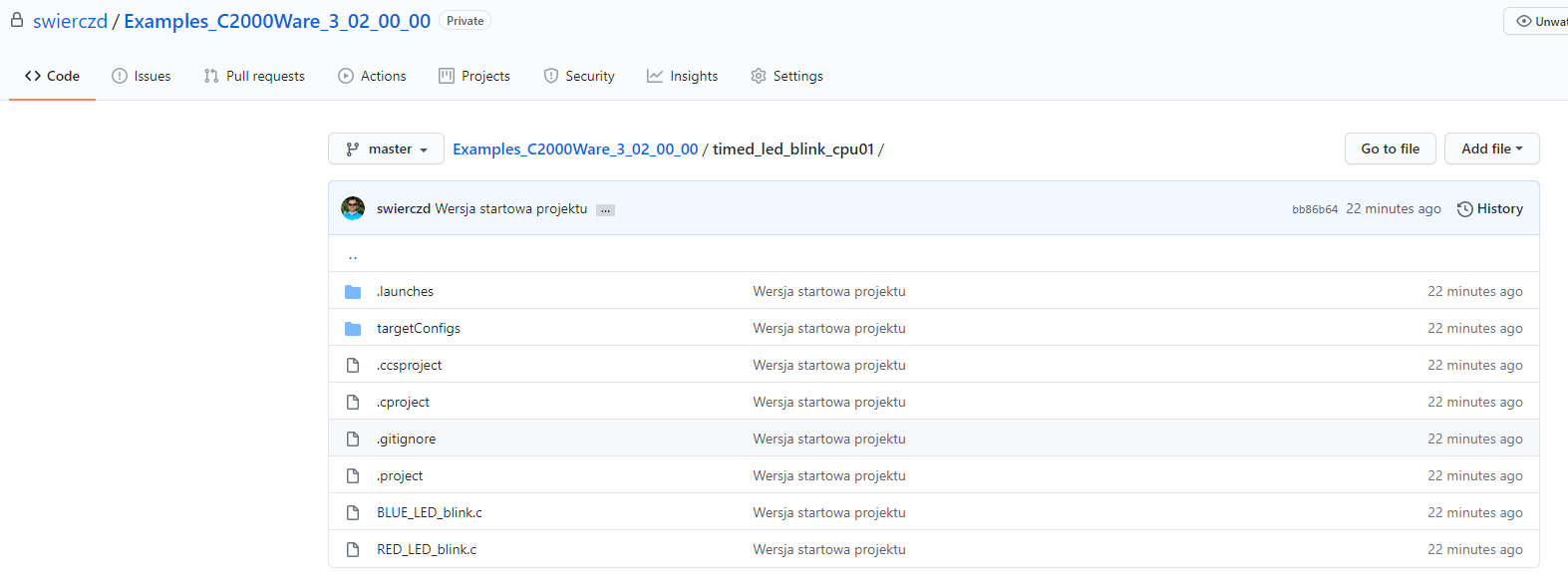


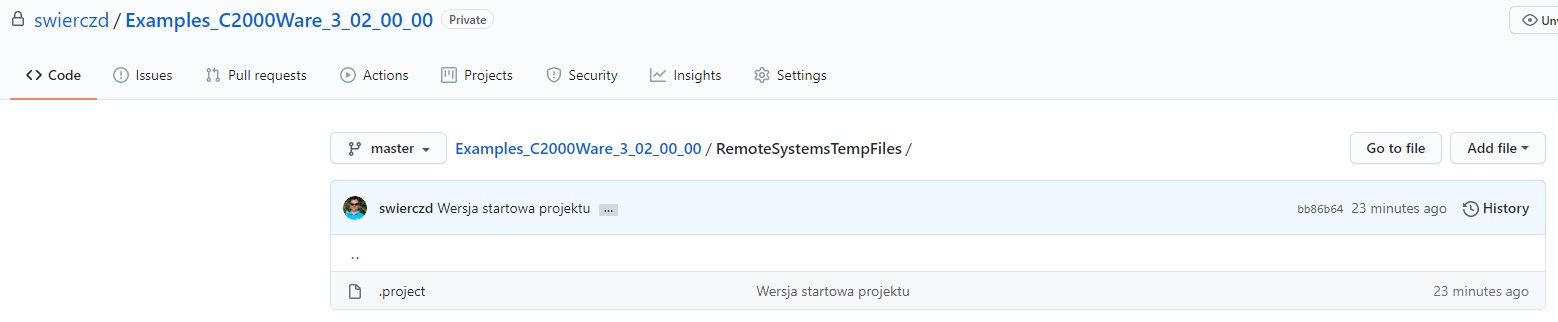


Na stronie konta GitHub mamy

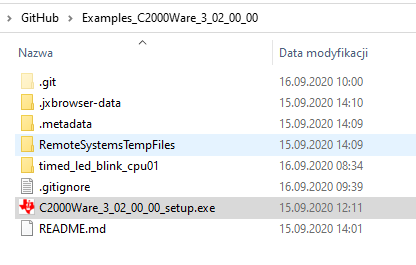




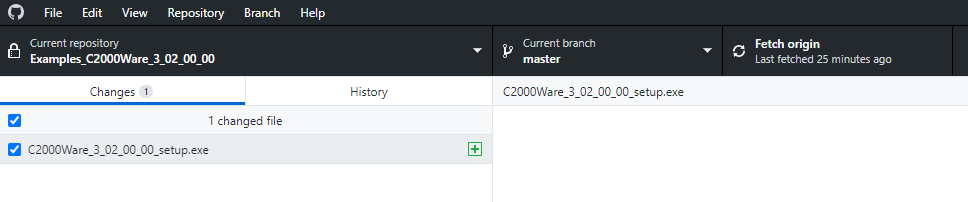




Do projektu dołączymy plik instalacyjny z C2000Ware w wersji 3\_02\_00\_00. Kopiujemy go do katalogu

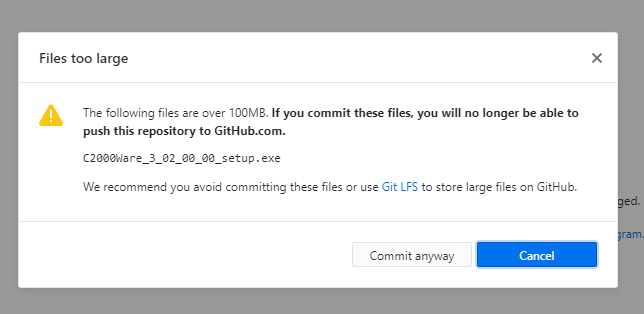


Git Hub zaraz zobaczy zmiany

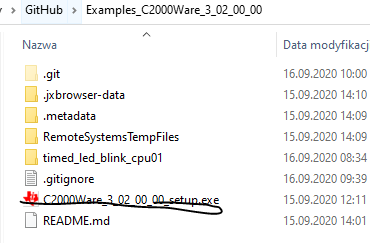


Musimy zrobić commita aby plik znalazł się na serwerze

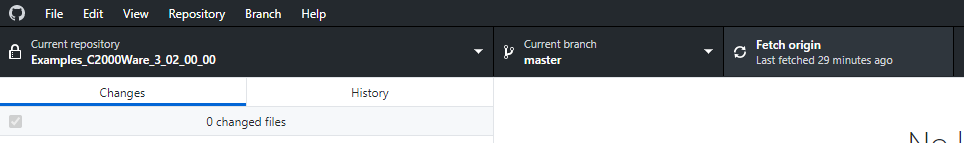
Po próbie uzyskujemy komunikat



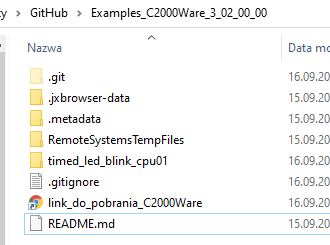
Robimy CANCEL. Usuwamy z katalogu

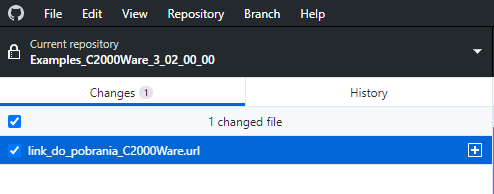


GitHUb nie widzi zmian

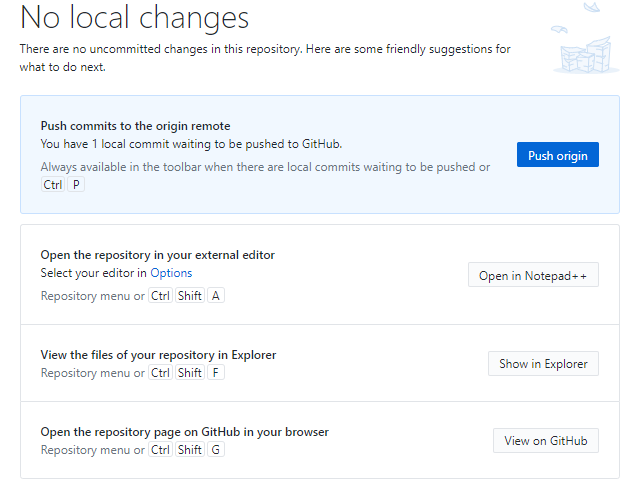


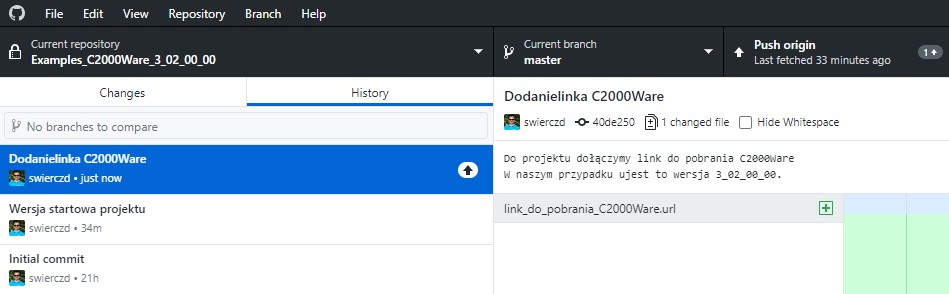
Zamiast dodawnia programu instalacyjnego C2000Ware dodamy link do pobrania oprogramowania



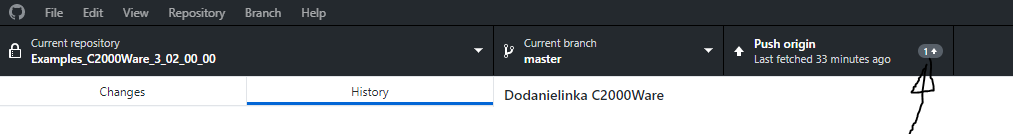


Robimy commit





Mamy do wysłania 1 plik



Na serwerze

