Dokumentacja projektu – czytnik kodu morse'a przy użyciu sygnałów świetlnych

Wykonawca : Sławomir Tenerowicz

Data : 25.01.2021

1. Niezbędne moduły do korzystania z programu:

- moduł czujnika światła, podpięty do portu **PTA0**

Komunikacja odbywa się za pomocą terminala **UART** ustawionego na częstotliwość **28 800** bitów na sekundę. Lista komend jest dostępna z terminala. Została również opisana poniżej.

Do testowania programu przydany może być **program Latarka**, który umożliwia wysyłanie kodów Morse z zadaną częstotliwością:

https://play.google.com/store/apps/details?id=pl.acform.android.torch&hl=pl&gl=US

2. Opis działania programu:

Program bazuje na przetworniku ADC wykorzystanego podczas ćwiczenia 5 w wersji ulepszonej o uśrednianie wyników przy pomocy licznika PIT.

Wykonywanie programu zostało podzielone na 4 etapy. Każdy z etapów jest aktywowany odpowiednią mu flagą (np. etap_0 = 1 oznacza aktywny etap zerowy). Wszystkie znajdują się w pętli głównej programu:

- **etap_0** : od niego domyślnie zaczyna się program. W trakcie jego trwania możliwe jest korzystanie z komend do wyświetlania i zmieniania wartości pewnych parametrów. Oto lista komend:

commands – wyświetla informacje o dostępnych komendach wraz z opisem działania.

vlt show – wyświetla napięcie aktualnie panujące na czujniku światła.

vlt set high *** - ustawia próg napięcia wysokiego, wartości napięcia odczytane z czujnika **większe** od tej wartości zostaną uznane za stan wysoki **(H).** Domyślnie ustawiona na 600 [mV]. W miejsce gwiazdek należy wpisać liczbę z przedziału 0-2900.

vlt set low *** - ustawia próg napięcia niskiego, domyślnie 400 [mV], wartości napięcia odczytane z czujnika światła **mniejsze** od tej wartości zostaną uznane za stan niski **(L).**

(znaczenie stanów wysokich i niskich zostaną opisane przy omawianiu etapu_1)

pit set period ** - ustawia okres wyzwalania przerwania na liczniku PIT [ms]. Co ten okres będzie pobierany uśredniony wynik z przetwornika ADC. W miejsce gwiazdek należy wpisać wartość okresu, domyślnie 10 [ms]

pit show – pokazuje aktualne ustawienie okresu licznika PIT

running config – wyświetla aktualną konfigurację programu

morse run – kończy etap_0, rozpoczyna odczytywanie sygnałów kodu morse'a

morse debug – umożliwia podgląd surowych danych i proces ich przetwarzania w wyniku którego otrzymano ciąg znaków.

- **etap_1**: Następuje zliczanie poziomów niskich (L) i wysokich (H). Co okres zerowania licznika PIT następuje uśrednianie odczytu napięcia panującego na czujniku światła. Wartość ta jest porównywana z:

- referencją stanu niskiego (*float stan_L*), ustawioną przy pomocy ***vlt set low*** (domyślnie 400 mV), jeśli jest mniejsza od tej wartości, inkrementowana jest zmienna *ile_stanow_L* w funkcji *przechwytywanie_sygnałów(),*

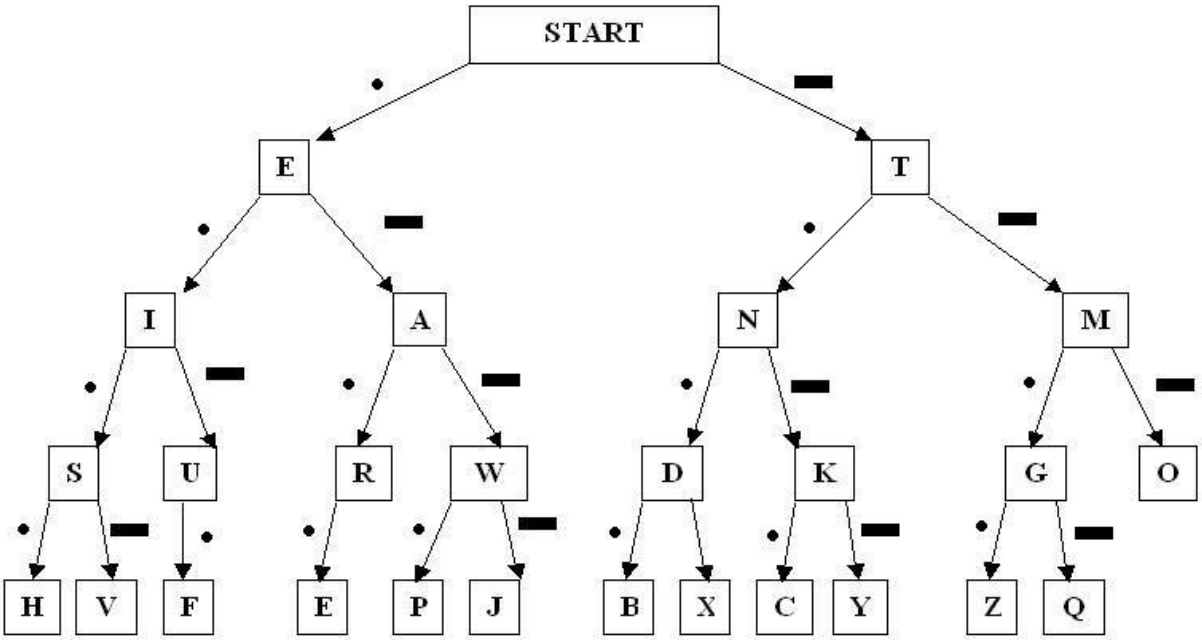
- referencją stanu wysokiego (*float stan_H*), jeśli odczytana wartość jest większa niż podana referencja, zwiększana jest liczba stanów wysokich zapisana w zmiennej *ile_stanow_H* w funkcji *przechwytywanie_sygnalow()*

Ilość stanów L i H jest zapisywana w tablicy *tablica_sygnalow*. Jeżeli stan H nie wystąpił w czasie trwania 10 kropek, uważa się że nastąpił koniec transmisji. Funkcja *przechwytywanie_sygnalow()* zwraca liczbę przechwyconych stanów.

Następuje etap 2.

-**etap_2** : Na tym etapie zostaje zdefiniowana długość kropki, czyli ilość zliczeń stanów L bądź H która odpowiada kropce. We wszystkich przypadkach, prócz tych gdzie ciąg znaków zaczyna się na literę 'e' bądź 't', kropka znajduje się na 2 miejscu (licząc od 0) w *tablicy_sygnalow*. Na jej podstawie generowana jest długość kreski (potrójna wartość kropki) oraz spacji (siedmiokrotność czasu trwania kropki). Oznaczenia liczbowe kreski, kropki, spacji zostały opisane w komentarzach w pliku *morse.c* . W momencie określenia wszystkich stanów jako kreskę bądź kropkę (bądź odstęp między kolejnymi wyrazami) i zapisania ich w *tablica_stanow*, zostaje wywołany etap 3.

- **etap_3** : Na podstawie *tablica_stanow* zostaje odtworzony kod morse'a. Idea jest bardzo prosta i bazuje na drzewku kodów morse'a.



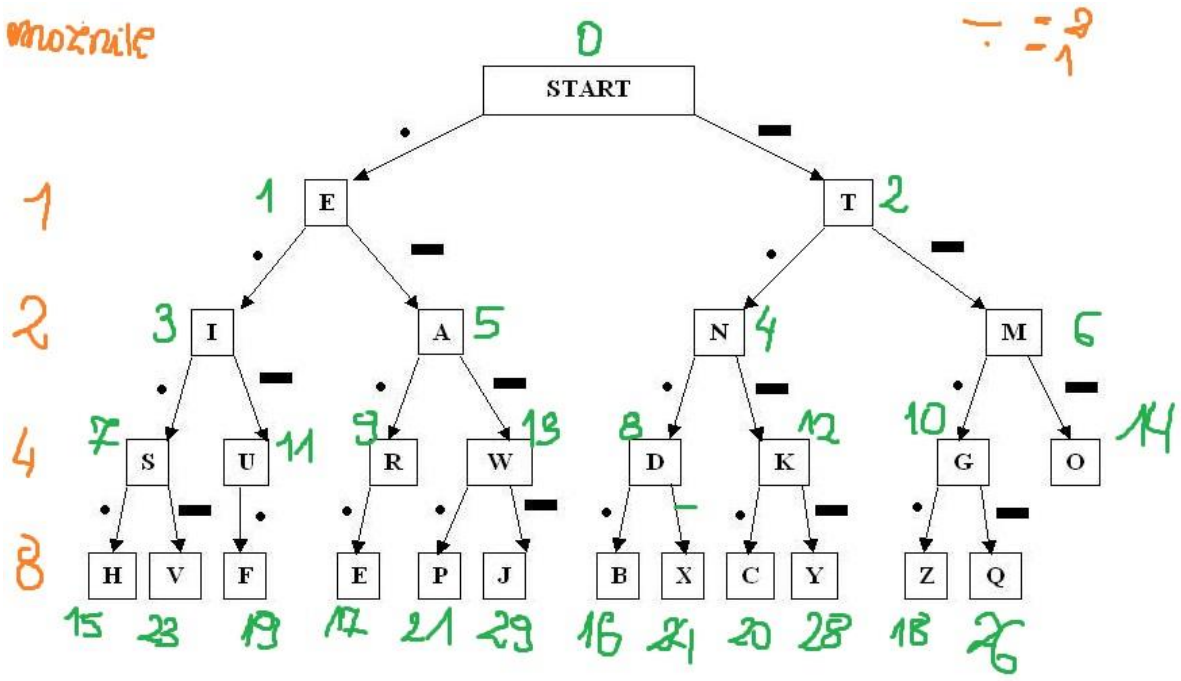
Rys 1. Drzewko kodów morse'a

Kresce przypisano wartość liczbową **2**, natomiast **kropce** przypisano **jedynkę**. W miarę schodzenia w dół po gałęziach drzewa, wartość kreski i kropki wzrasta dwukrotnie za każdy poziom (na pierwszym poziomie – gdzie występuje tyko 'e' i 't' mnożnik wynosi 1, na drugim poziomie 2, na trzecim 4 a na czwartym 8). W ten sposób każdej literze przypisano wartość numeryczną, stworzoną na podstawie kreski i kropki, co z kolei na podstawie kreski i kropki pozwoli odtworzyć wartość numeryczną litery jak i samą literę. Przykładowo dla litery F mamy sekwencję (idąc od góry) ···- co daje nam:

$$'\cdot' = 1 \qquad '-' = 2$$

$$F = \cdot_1 \cdot_2 -_3 \cdot_4 = 1 \cdot 2^1 + 1 \cdot 2^2 + 2 \cdot 2^3 + 1 \cdot 1^4 = 19$$

Przesuwając się po kolejnych wartościach tablicy stanów ustala się liczbę, a na jej podstawie dobiera się odpowiedni znak. Zdekodowanie całej tablicy stanów kończy etap_3. Następuje wypisanie ciągu znaków oraz powrót do etapu_0. Przy wybraniu komendy ***morse debug*** zostaną wyświetlone *tablice sygnałów i stanów* jak również odpowiadająca im *tablica znaków*. W przypadku ponownego wybrania ***morse run*** tablice te są czyszczone i następuje kolejny zapis.



Rys 2. Drzewko kodów morse'a – wartości liczbowe

Przypisanie kreski i kropki do pojedynczego sygnału odbywa się w pewnym obszarze niepewności – w tej wersji programu jest to na sztywno ustawione +-10%, choć w przypadku kodu morse'a generowanego przy pomocy programu **Latarka**, nie stanowi to problemu.

Ważne jest to, by okres licznika PIT był wystarczająco krótki – by móc zmierzyć tyle impulsów L lub H by jednoznacznie zdefiniować kreskę czy kropkę. Dla przykładu, jeśli okres licznika PIT wynosi 10 [ms] a długość kropki wynosi 300 [ms] to zostanie zarejesrtowanych około 30 impulsów w stanie H. Przy debugowaniu, kody dla których na kropkę przypadało 20 impulsów były dekodowane prawidłowo.