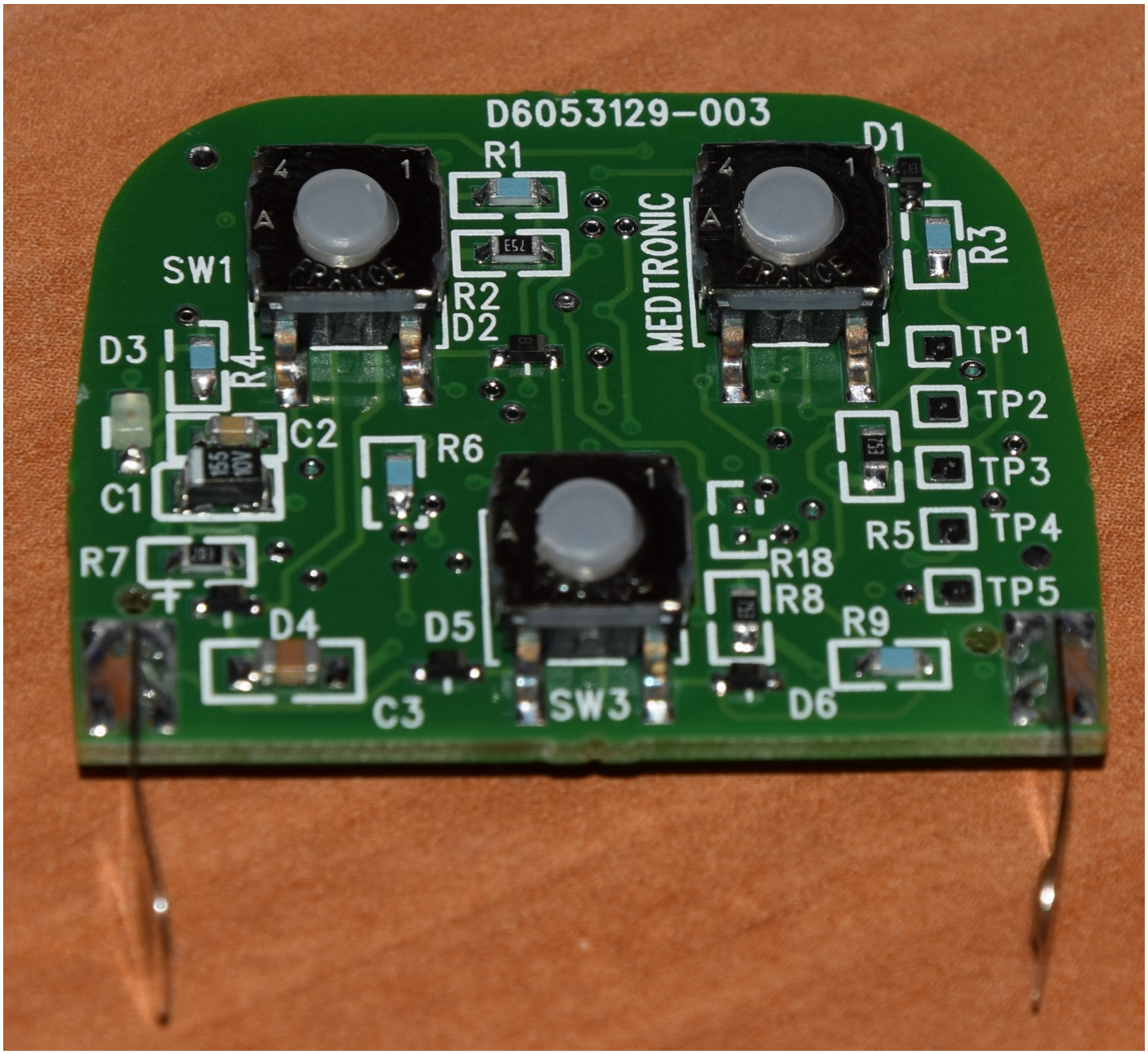
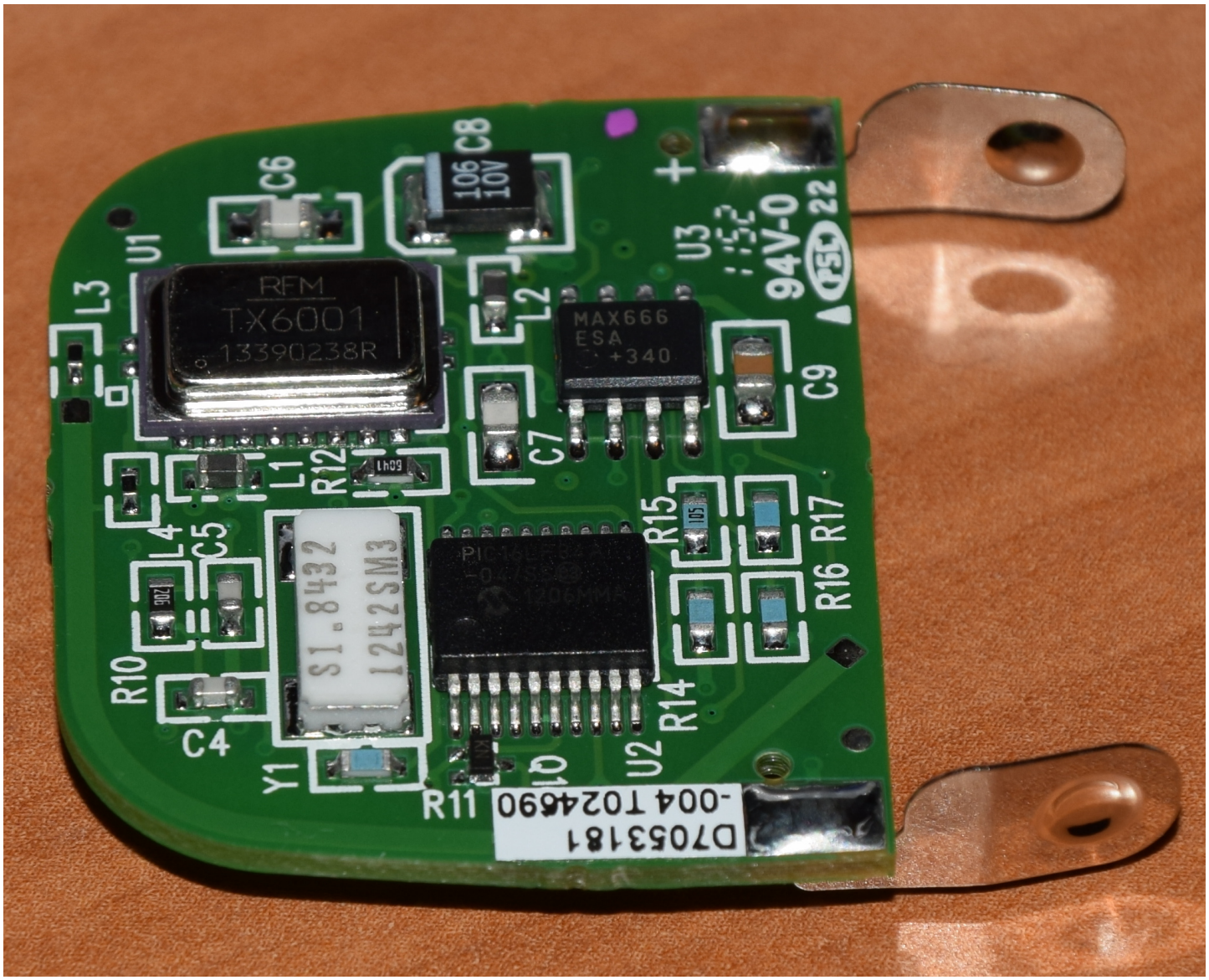
JAK ROZKODOWALIŚMY UNIWERSALNY PILOT DO POMP INSULINOWYCH MEDTRONIC PARADIGM TYP MMT-503EU

Wygląd pilota przypomina bardziej urządzenie do bramy niż coś, co miałoby sterować pompą insulinową:

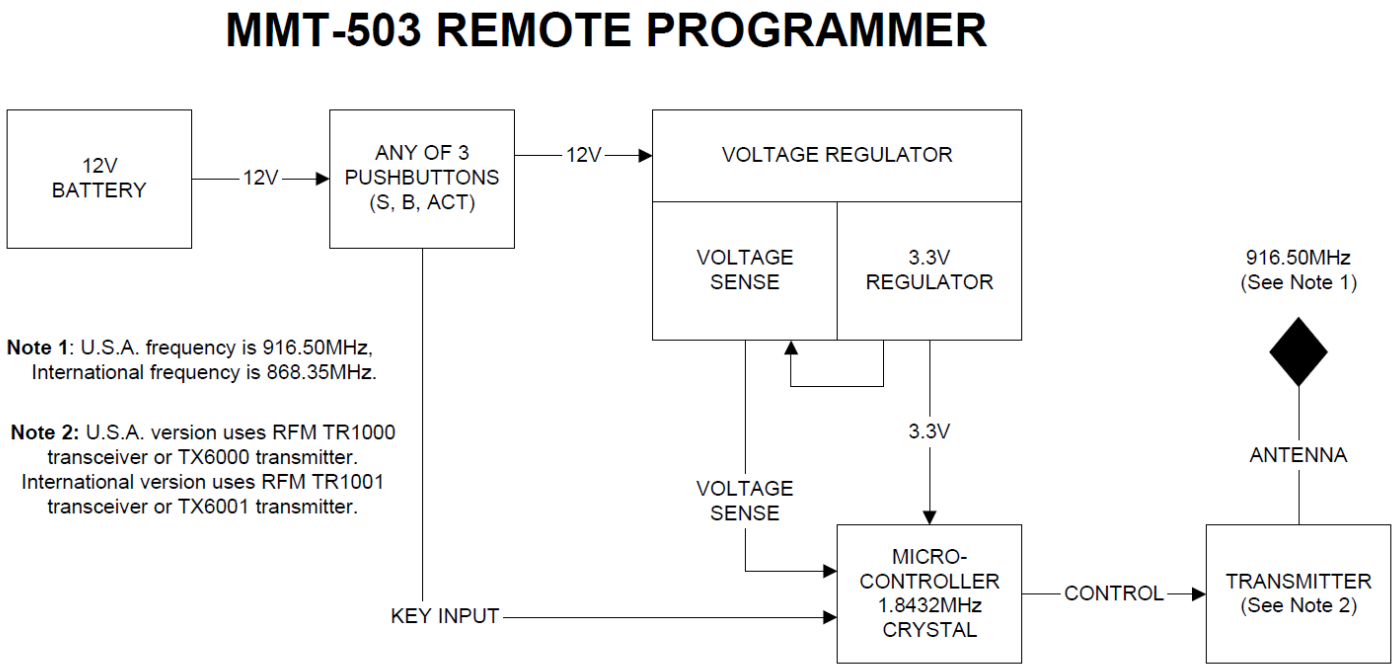
opracowanie: Sławomir Malinowski

Płytka pilota zawiera mikrokontroler PIC16LF84A, w którym zapisany jest kod programu oraz transmiter TX6001, który wysyła pakiety danych do pompy insulinowej. Oprócz tych dwóch strategicznych elementów jest jeszcze regulator napięcia MAX666. W pilocie zastosowano 12V baterię. Widok obu stron płytek:



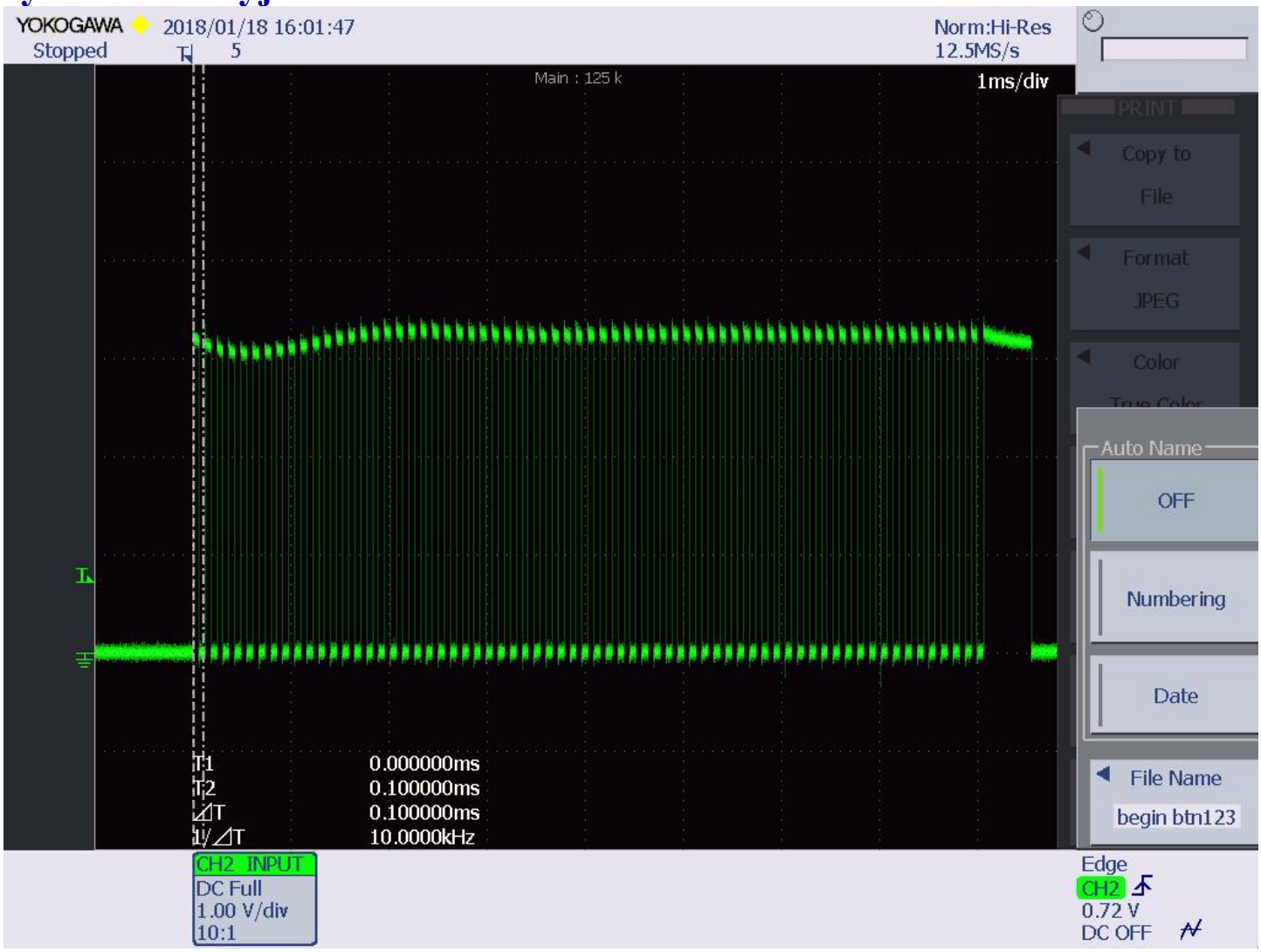

RFM
TX6001
13390238R
MAX666
ESA
+340
106
10V
C8
C6
U1
L3
L2
U3
94V-0
C9
C7
L1
R12
L4
C5
R15
R17
R16
R10
C4
R14
Y1
R11
U2
Q1
S1.8432
1242SM3
D7053181
-004 T0246 90

Dla naszego regionu transmisja odbywa się na częstotliwości 868MHz. Schemat blokowy:

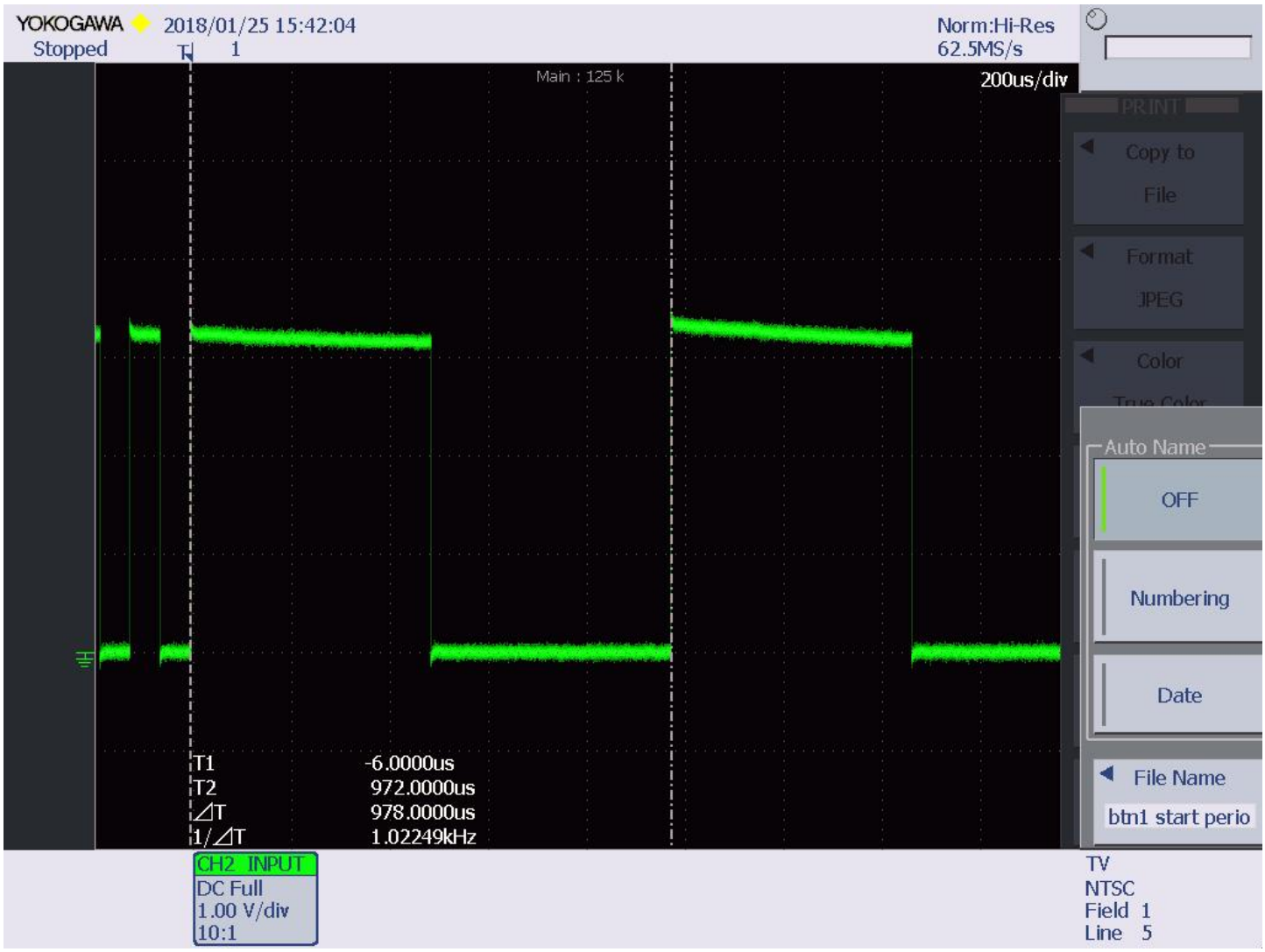


Pilot potrafi podawać tylko bolusy proste oraz zatrzymywać i wznawiać pracę pompy. W menu pompy wybiera się krok z jakim ustawiamy pojedynczymi naciśnięciami przycisku pilota wartość bolusa. Najmniejszy krok to 0,1U, następny możliwy to 0,5U. Ilość naciśnięć przycisku ustawia wartość bolusa. I tutaj niestety trzeba w głowie liczyć ilość naciśnięć a najlepiej ilość pików dźwiękowych z pompy. Po nieznaczym oddaleniu pilota od pompy, już nieco ponad 2 metry, niektóre kliknięcia nie docierają do pompy i wtedy nie mamy pewności ile w końcu się nabiło jednostek. W takim wykonaniu i działaniu pilot jest dla mnie nie do zaakceptowania. Zrodził się pomysł, aby na podstawie tego oryginalnego pilota zrobić zupełnie nowy pilot od podstaw ze wszelkimi możliwymi funkcjami również odbieranymi z pompy, jeśli projekt się powiedzie całkowicie to chcemy w nowym pilocie uwzględnić nawet CGM pobierany z pompy. Nowy pilot będzie oparty na zupełnie nowych podzespołach z wyświetlaczem graficznym z MENU, do obsługi będzie 5 przycisków, układ radiowy na transceiverze na pewno zwiększy zasięg działania od pilota do pompy. Po każdorazowym wykonaniu sterowania pompa będzie odpytana o aktualny status. Historia bolusów, poziom insuliny w zbiorniku, aktywna insulina będą dostępne na ekranie pilota.

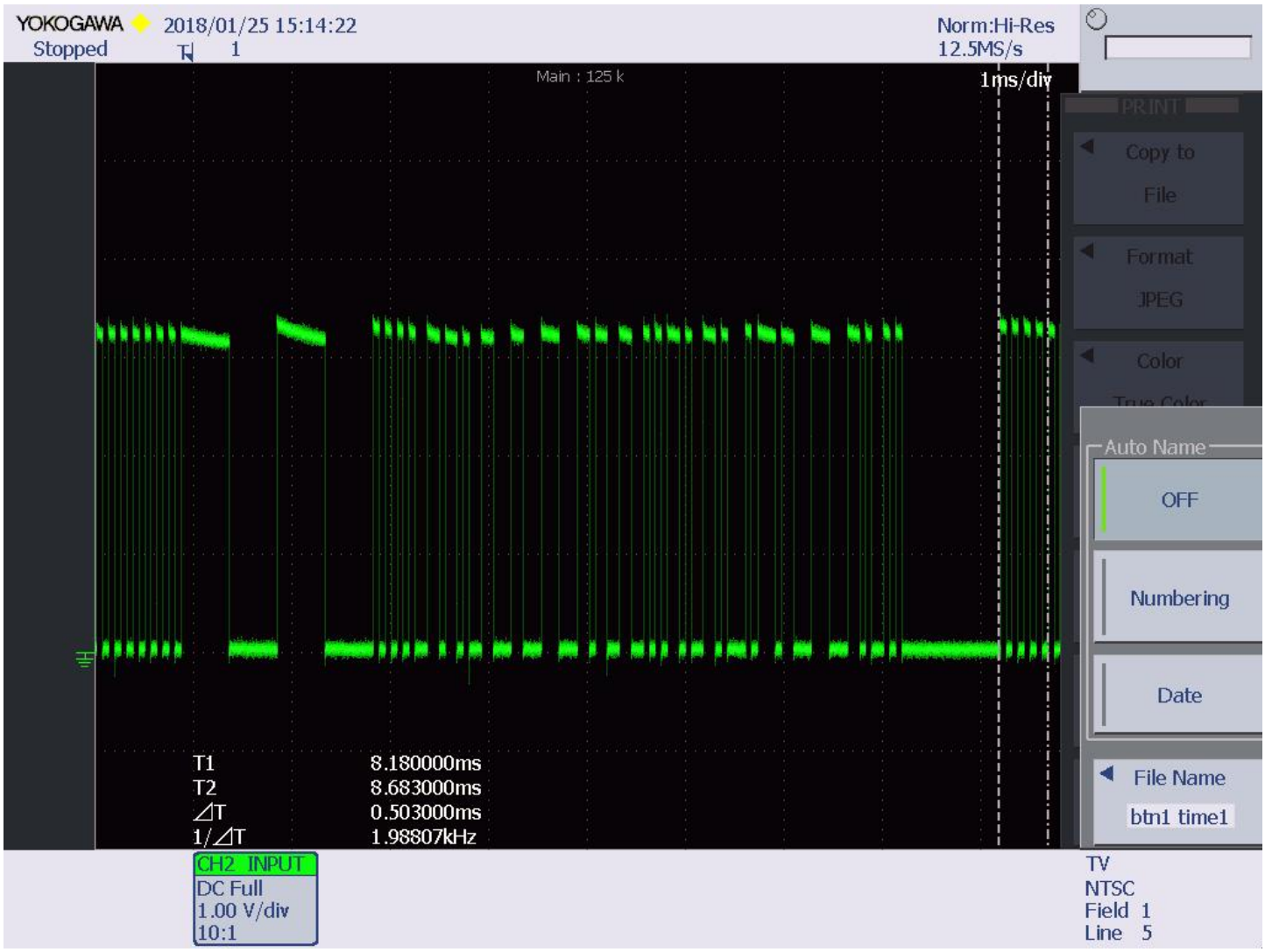
Jak zabraliśmy się do zdekodowania programu pilota i na czym to polegało? Zastosowana metoda to tzw. inżynieria wsteczna, tutaj polegało to na tym, aby na podstawie fali prostokątnej z wyjścia mikrokontrolera do transmitera rozbić sekwencje dla wielokrotnych naciśnięć wszystkich trzech przycisków. Zebrane paczki danych z oscyloscopu trzeba było podzielić na stałą i zmienną część. Okazało się, że sygnał zmienia się również za każdą sekwencją wysłaną przy naciskaniu tego samego przycisku. Przytrzymanie przycisku generuje tą samą sekwencję bez zmiany danych. Trzeba było znaleźć i wydzielić z wysyłanego pakietu impulsy startowe i synchronizacji, impulsy kończące każdą sekwencję oraz najważniejsze w jaki sposób wysyłany jest numer seryjny pilota do pompy, rozróżnianie naciśniętych przycisków oraz rozwiązać zagadkę w jaki sposób zmieniają się klucze z każdym kolejnym cyklem. Jak się później okazało jest to pilot z kodem zmiennym z dodatkowo zastosowanym szyfrowaniem w rodzaju 4b6b. Na kolejnych stronach zamieszczam screeny z oscyloskopu dla przykładowej sekwencji danych. Początkowa część synchronizacyjna:



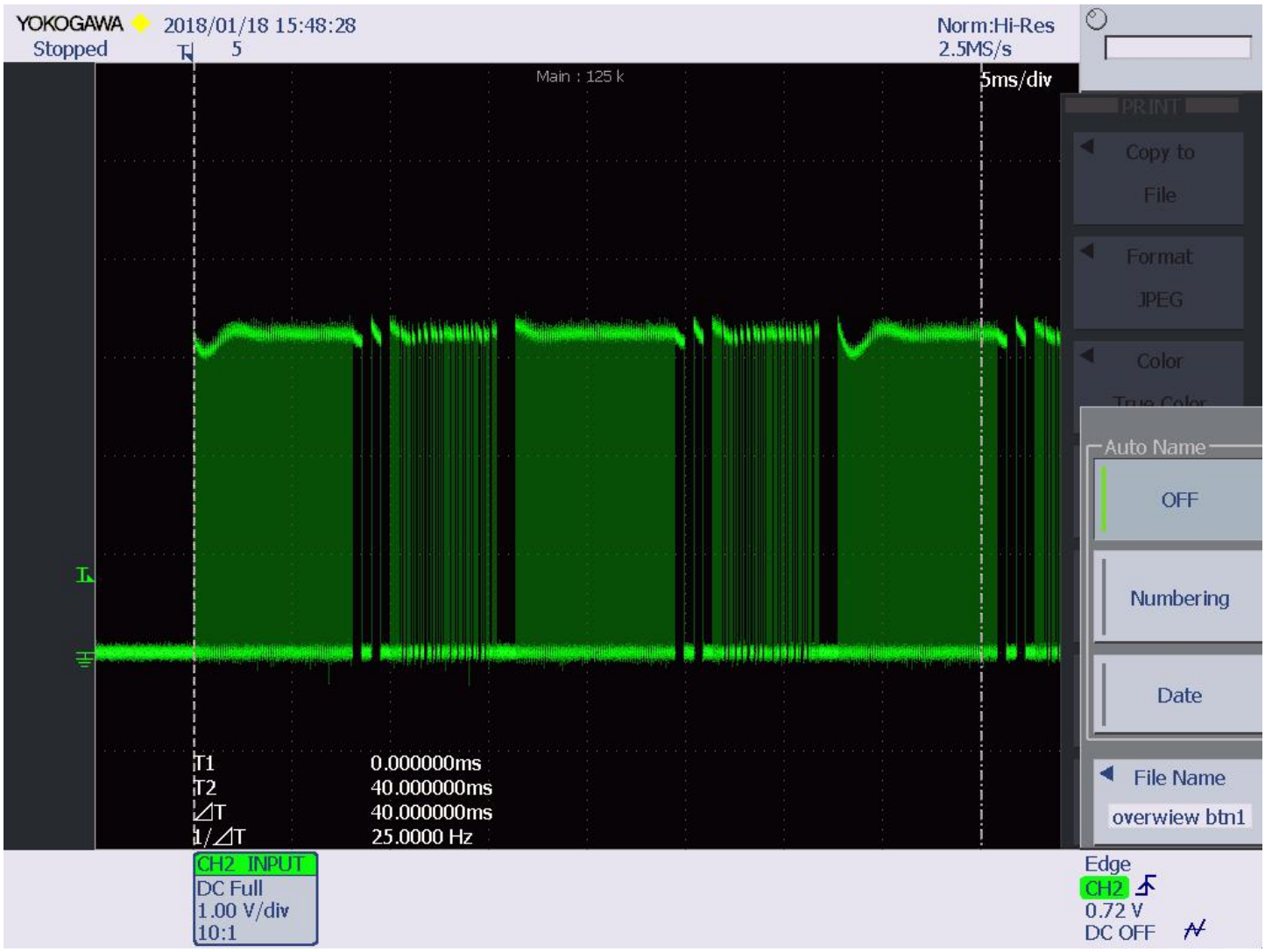
Następnie część startowa:



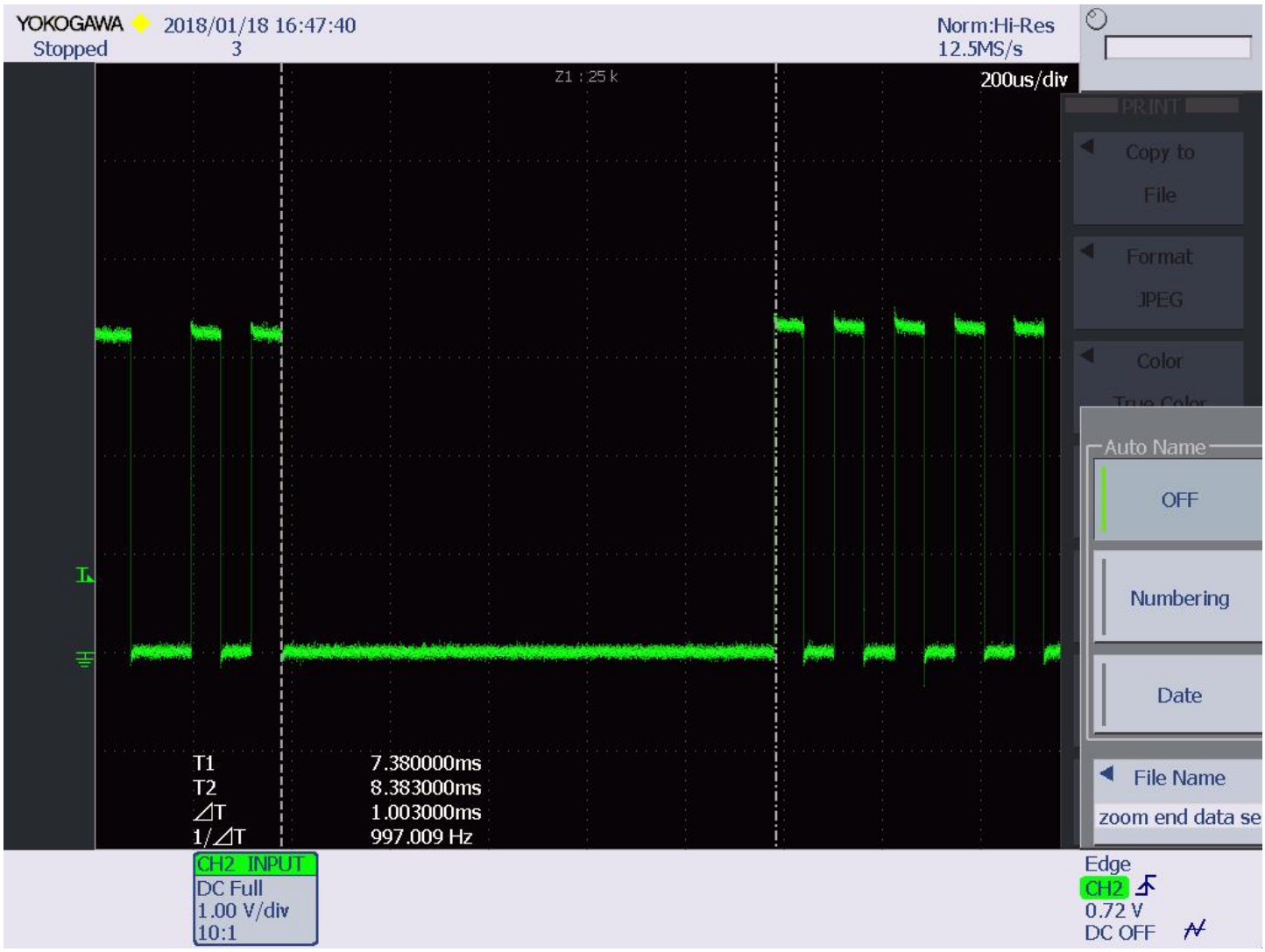
Dalej właściwa sekwencja danych z impulsami kończącymi pakiet:



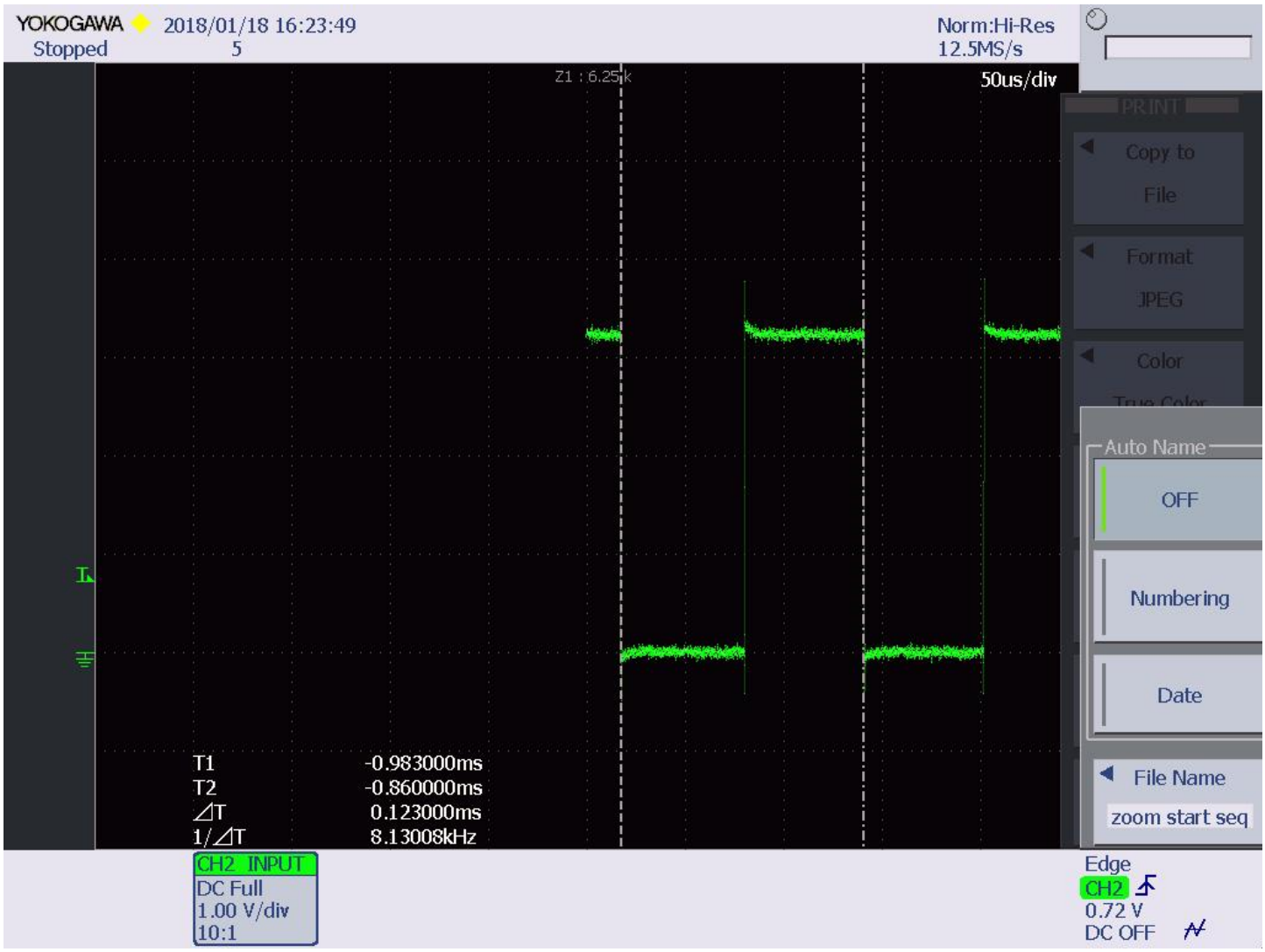
Ciąg kilku sekwencji po przytrzymaniu przycisku:



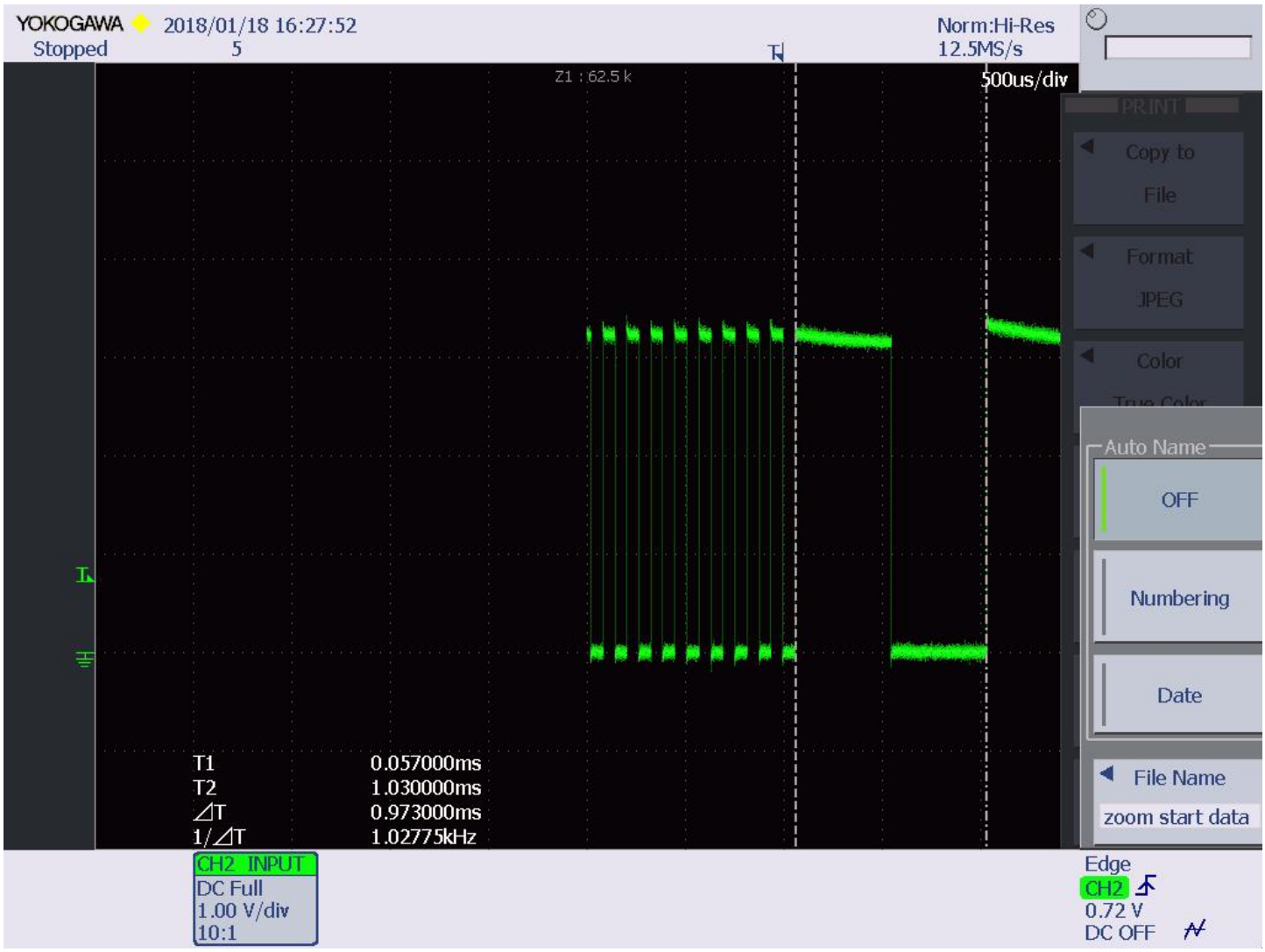
Obliczenia timingów pomiędzy sekwencjami:



Obliczenia timingów części synchronizacyjnej (Preambuły):



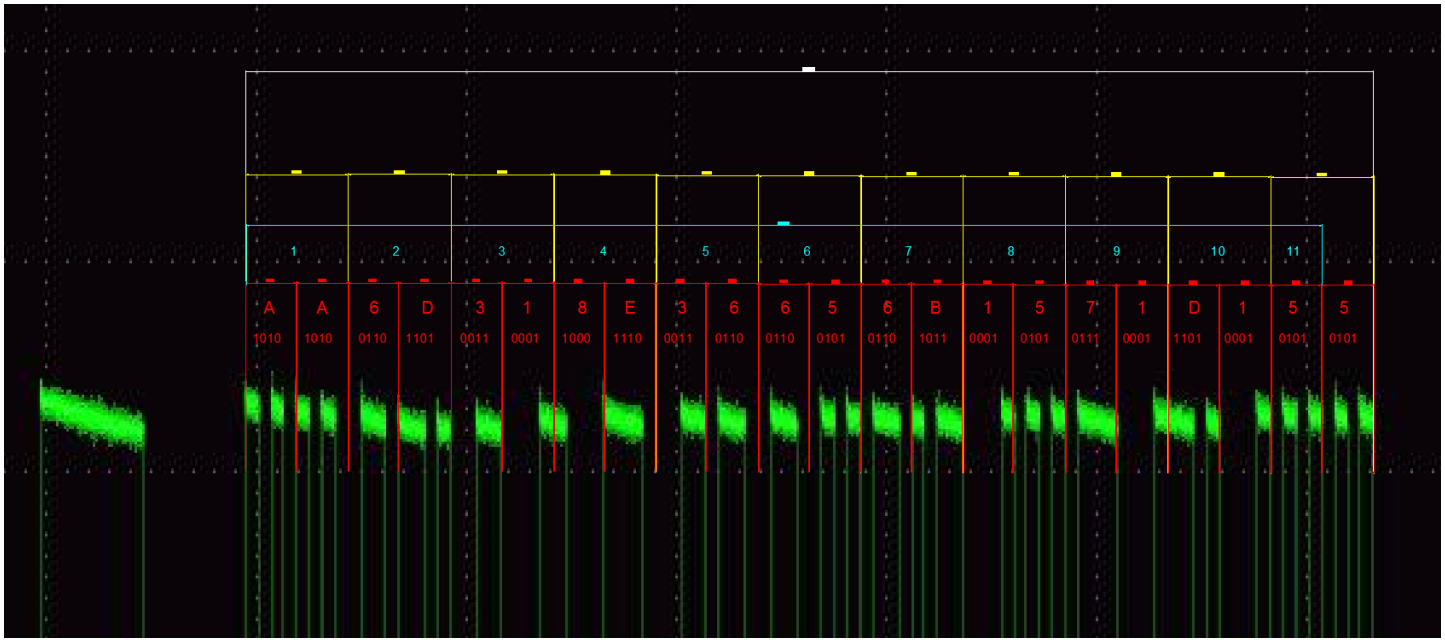
Obliczenia timingów części startowej (Sync Word):



Kompletna ramka danych zawiera 33,5 bajta.
Początkowa część synchronizacyjna to 16,5 bajta bez szyfrowania.
Część startowa sekwencji to 4 bajty bez szyfrowania.
Początek sekwencji danych to 0,5 bajta bez szyfrowania.
Typ urządzenia zdalnego sterowania to 1 bajt bez szyfrowania.
Indywidualny numer seryjny pilota to 4,5 bajta w kodzie 4b6b
Numer naciśniętego przycisku to 1,5 bajta w kodzie 4b6b
Wartość zmienna sequence # to 1,5 bajta w kodzie 4b6b
Suma kontrolna CRC-8 to 1,5 bajta w kodzie 4b6b
Końcowa sekwencja ramki danych to 2,5 bajta bez szyfrowania.

Cała ramka musiała zostać podzielona na pojedyncze bajty, następnie każdy bajt musial zostać przeliczony na wartość hexadecymalną i wstawiony do tablicy LUT. Najpierw została stworzona tablica LUT dla każdego przycisku osobno bez szyfrowania danych. Kolejny etap to było utworzenie docelowej tablicy LUT z uwzględnieniem kodowania 4b6b.

Jak wygląda wydzielanie poszczególnych bitów z fali prostokątnej:



Dokładna analiza całego pakietu danych, nie wliczając bajtów synchronizacji i bajtów końcowych pełnej ramki pojedynczego cyklu:

bajty								hex	
1	1	0	1	0	1	0	1	0	A	A
2	0	1	1	0	1	1	0	1	6	D
3	0	0	1	1	0	0	0	1	3	1
4	1	0	0	0	1	1	1	0	8	E
5	0	0	1	1	0	1	1	0	3	6
6	0	1	1	0	0	1	0	1	6	5
7	0	1	1	0	1	0	1	1	6	B
8	0	0	0	1	0	1	0	1	1	5
9	0	1	1	1	0	0	0	1	7	1
10	1	1	0	1	0	0	0	1	D	1
11	0	1	0	1	0	1	0	1	5	5

słowa 6-bitowe						
1	1	0	1	0	1	0
2	1	0	0	1	1	0
3	1	1	0	1	0	0
4	1	1	0	0	0	1
5	1	0	0	0	1	1
6	1	0	0	0	1	1
7	0	1	1	0	0	1
8	1	0	0	1	0	1
9	0	1	1	0	1	0
10	1	1	0	0	0	1
11	0	1	0	1	0	1
12	1	1	0	0	0	1
13	1	1	0	1	0	0
14	0	1	0	1	0	1

packet data w kodzie 4b6b	
A 6	DEVICE TYPE
4 1 3 3 9 5	RF SERIAL NUMBER
8 1	COMMAND BUTTON
0 1	SEQUENCE #
4 0	CRC-8

1	1	0	1	0	startowy półbajt pakietu, zaliczany do zdekodowania w 4b6b, podlega wyliczeniu CRC-8
	0	1	0	1	końcowy półbajt pakietu, nie zaliczany do zdekodowania w 4b6b, nie podlega wyliczeniu CRC-8

CRC-8 liczone jest z tablicy {0xA6, 0x41, 0x33, 0x95, 0x81, 0x01} przy zastosowaniu wielomianu 0x9B

gdzie:
A6 to device ID (pilot MMT-503EU),
0x41, 0x33, 0x95 to numer pilota 41 33 95 czyli **413395**,
0x81 kod komendy (naciśniętego przycisku), pozostałe dwa to 0x86 i 0x88,
0x01 – sequence #, ilość naciśnięć przycisku, inkrementacja (0x0 ... 0xFF),
Po policzeniu CRC doklejane jest na koniec tej tablicy i potem całość konwertowana jest na kod 4b6b.