Projekt komputera opartego o mikroprocesor 6502 i układy z serii 74xx z obsługą monitora VGA oraz klawiatury PS/2

Technika Cyfrowa 2022

Norbert Morawski

12 kwietnia 2022

Spis treści

1 Wstęp **2**

2 Moduły **2**

3 Moduł zegarowy **2**
3.1 Budowa 3

4 Karta graficzna **4**
4.1 Funkcje 5
4.2 Schemat blokowy 6

5 Kontroler klawiatury PS/2 **7**
5.1 Funkcje 7

6 Jednostka centralna **8**
6.1 Możliwości 8
6.2 Mapa pamięci 10
6.3 Przykłady działania 11

1 Wstęp

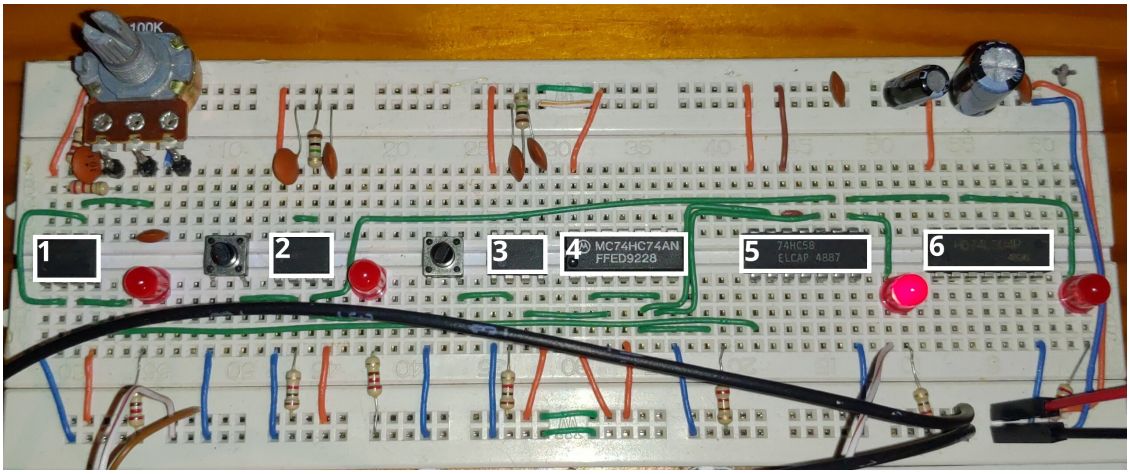
Inspiracją do zbudowania komputera były filmy na platformie YouTube autorstwa pana Bena Eatera.

2 Moduły

Komputer składa się z czterech głównych modułów. Są to:

- Moduł zegarowy oparty o układy **NE555**,
- Moduł karty graficznej **VGA**,
- Moduł kontrolera klawiatury **PS/2**,
- Moduł główny z mikroprocesorem **MOS Technology 65C02**.

3 Moduł zegarowy



Rysunek 1: Moduł zegarowy

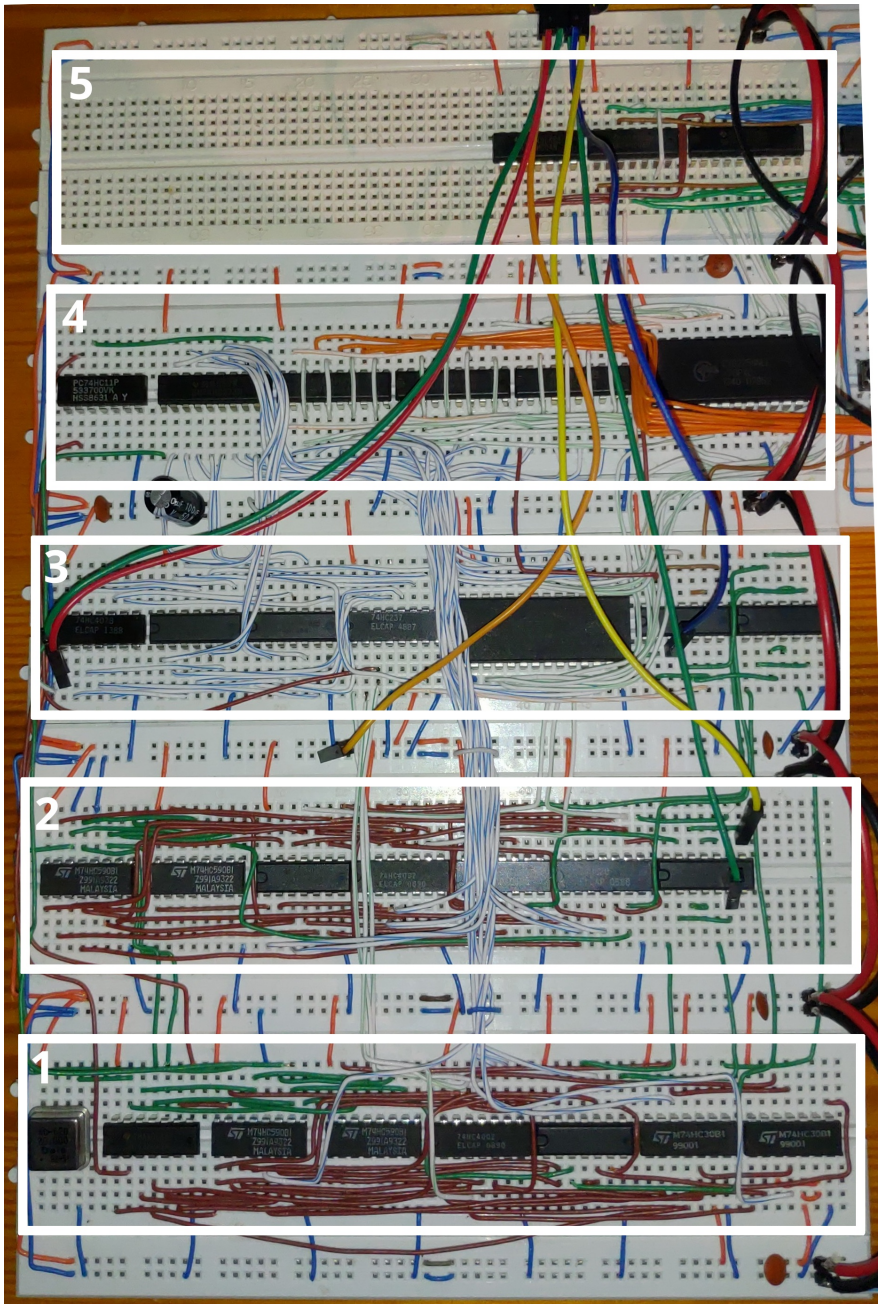
Zespół ten składa się z:

1. **NE555** – układ generujący przebieg prostokątny
2. **NE555** – układ zapobiegający drganiom styków przycisku manualnego taktowania
3. **NE555** – układ zapobiegający drganiom styków przycisku zmiany trybu
4. **74HC74** – przerzutnik D; pamięć trybu
5. **74HC58** – dwie bramki AND i jedna OR; wybór trybu
6. **74LS04** – inwerter

3.1 Budowa

Oparty o trzy układy czasowe **NE555** moduł zegarowy jest w stanie generować przebiegi prostokątne o częstotliwości do 153800Hz. Moduł ten może pracować w dwóch trybach, tj. automatycznym z generacją częstotliwości przez jeden z układów czasowych, jak również manualnym, gdzie użytkownik wciskając przycisk wywołuje powstanie jednego taktu zegara. Funkcja manualnego zadawania przebiegu zegarowego była przydatna we wczesnych etapach pracy nad projektem w celu usuwania błędów oprogramowania i sprzętu przy pomocy analizatora stanów logicznych opartego o układ **Atmega 32**.

4 Karta graficzna



Rysunek 2: Moduł karty graficznej VGA

Moduł karty graficznej składa się z następujących pod-modułów:

1. Licznik pikseli oparty o dwa układy **74HC590** z zegarem 20MHz,
2. Licznik linii oparty o dwa układy **74HC590**, dwa zatrzaski typu RS, służące do generacji sygnałów synchronizacji poziomej (zielony przewód) i pionowej (żółty przewód),

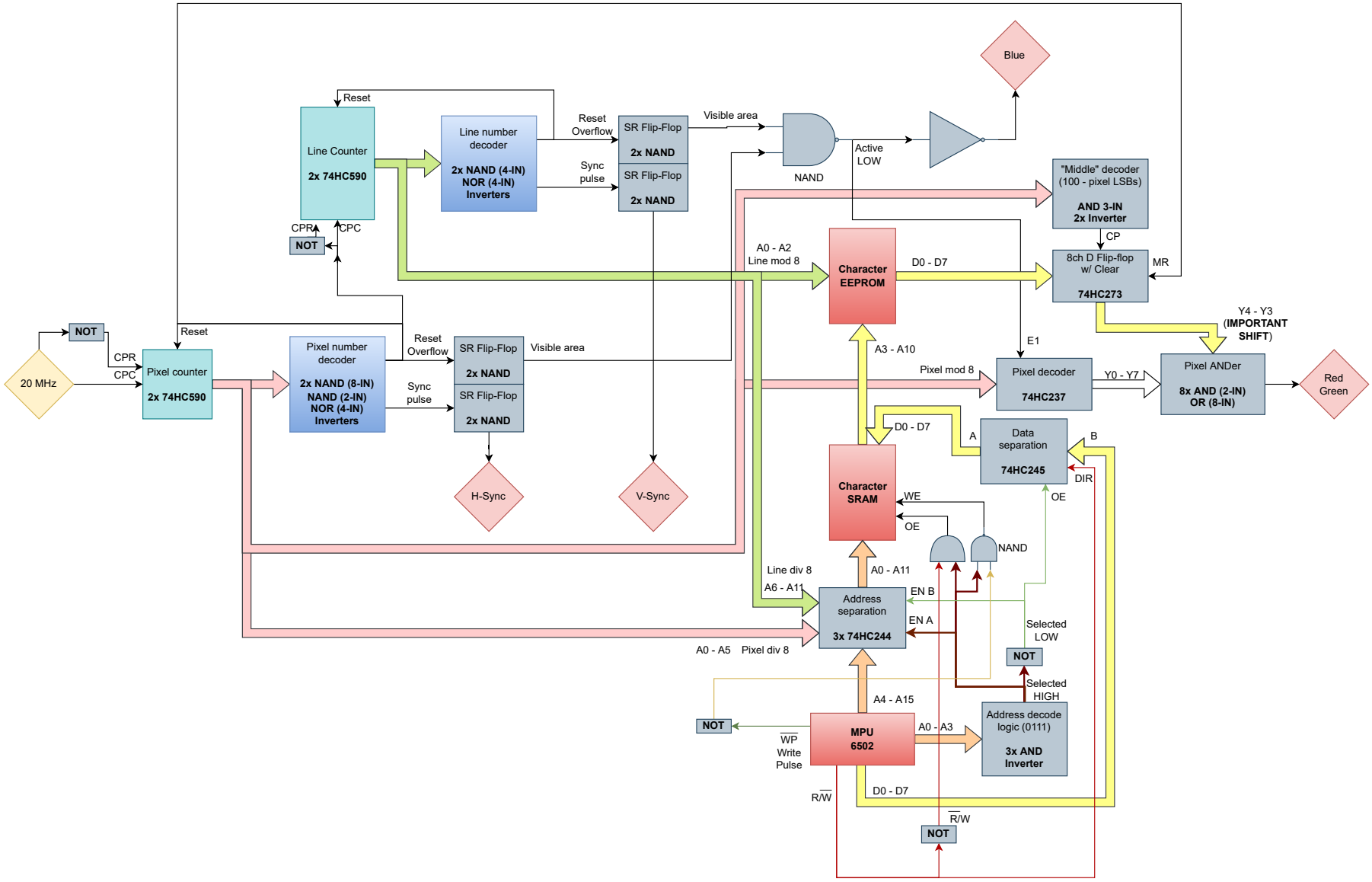
3. Dwa zatrzaski typu RS z bramką NAND służące do generacji niebieskiego tła tylko w części widocznej ramki obrazu VGA, generator znaków EEPROM, dekoder pikseli,
4. Pamięć znaków SRAM, bufory szyny adresowej, 8-bitowy przerzutnik D przechowujący aktualnie wyświetlaną linię 8 pikseli,
5. Bufory szyny danych, logika dekodująca adres karty pamięci

4.1 Funkcje

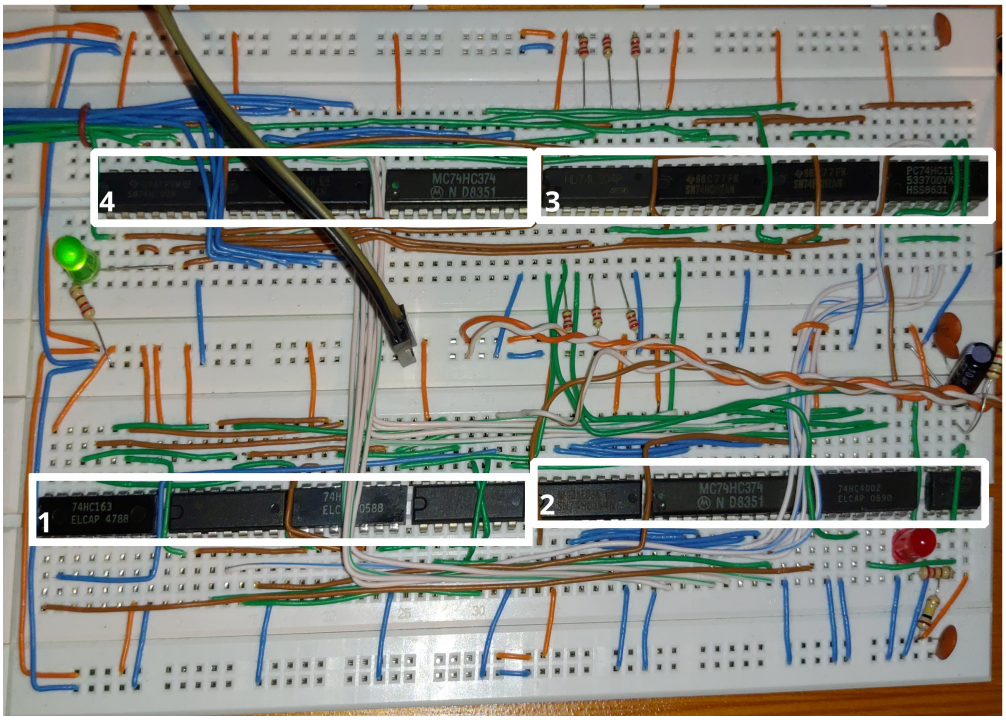
Karta obsługuje tryb znakowy 49 x 37 znaków. Wynika to z obsługi 1/4 rozdzielczości 800x600 tj. 400x300 i znaku o rozmiarze 8x8 pikseli. Połowa znaku na początku linii jest poświęcona na dostęp do pamięci RAM+EEPROM, gdzie po 4 pikselach, po ustabilizowaniu się danych, aktualna linia jest zatrzaskiwana do przerzutnika D w sekcji czwartej. Do pamięci EEPROM została załadowana strona kodowa 437. Użyta została czcionka IBM-CGA.

Dostęp do pamięci karty realizowany jest poprzez odcięcie liczników i pamięci generatora znaków co objawia się krótkimi artefaktami (widocznymi podczas czyszczenia ekranu na załączonych filmach). Metoda ta została wybrana w celu uproszczenia implementacji i sprzętu. Istnieje pole do optymalizacji. Np. mikroprocesor mógłby w przerwaniu wpisywać znaki z bufora do pamięci karty lub należałoby zastosować pamięć dwuportową.

4.2 Schemat blokowy



5 Kontroler klawiatury PS/2



Rysunek 3: Moduł kontrolera klawiatury

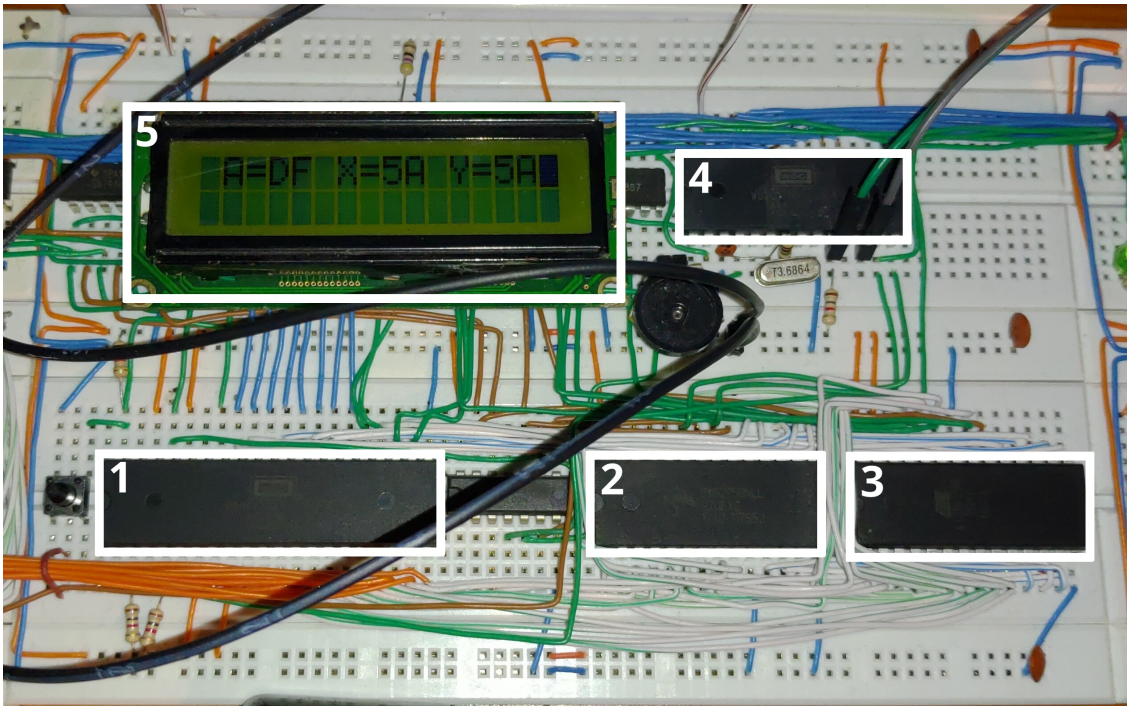
Funkcjonalność odczytu klawiszy z klawiatury zapewniają następujące układy:

1. Licznik liczący bity i aktywujący rejestr przesuwny w celu zapisania do niego tylko 8 bitów kodu klawisza,
2. Rejestr przesuwny z przerzutnikiem D przechowującym odebrany kod klawisza, układ resetu,
3. Układ dekodujący kody`E0` i `F0` w celu wykrywania kodów rozszerzonych i podnoszenia klawisza,
4. 8-bitowy przerzutnik D; rejestr stanu, zawiera bity odpowiedzialne za wystąpienie przerwania klawiatury, wystąpienie kodu E0, F0, bufor magistrali danych, inwerter.

5.1 Funkcje

Układ automatycznie odczytuje kody `E0` (*extended*) i `F0` (*break*), przez co mikroprocesor szybciej może wczytać dane i zareagować. Moduł generuje maskowalne przerwania na wejściu $\overline{\texttt{IRQ}}$ mikroprocesora. Po odczycie rejestru stanu przerwanie jest dezaktywowane. Układ resetuje się samoczynnie po włączeniu zasilania.

6 Jednostka centralna



Rysunek 4: Moduł mikroprocesora

Główny moduł składa się z:

1. **MOS Technology 65C02** – mikroprocesor; ulepszona wersja procesora 6502 stosowanego m. in. w komputerach takich jak Apple I/II czy Commodore 64,
2. **CY62256** – 32KB moduł (adresowalne 16KB) pamięci statycznej RAM,
3. **AT28C256** – 32KB moduł pamięci EEPROM,
4. **65C51N** – moduł komunikacyjny UART; umożliwia dynamiczne ładowanie oprogramowania bez przeprogramowywania pamięci stałej,
5. Wyświetlacz oparty o **HD44780**, podłączony bezpośrednio do szyny danych mikroprocesora.

6.1 Możliwości

Bootloader Program ładujący znajdujący się w pamięci EEPROM umożliwa dynamiczne ładowanie programów do pamięci RAM a także weryfikacje poprawności ich przesłania. Sprawdza on także dwa bajty startowe przed skokiem do aplikacji użytkownika w pamięci RAM. Fragment kodu czytającego jedną stronę (256 bajtów) z interfejsu szeregowego i zapisującego tę stronę do pamięci RAM pod wskazany adres.

```
write_page:
  ; Writes a page
```

```
  ; Needs to be sent a page address (1 byte)
  ; and 256-byte page data

  pha
  phy

  ; Receive page address
  jsr uart_recv

  ; Setup page write address
  stz BTLDR_PAGE_ADDRESS        ; Store low byte first (always zero)
  sta BTLDR_PAGE_ADDRESS + 1    ; Store high byte
  ldy #0                        ; Clear Y index

.loop:
  jsr uart_recv
  sta (BTLDR_PAGE_ADDRESS), y

  iny
  bne .loop
  ; End loop

  ply
  pla
  jmp program_loop
```

Interfejs szeregowy UART Oprócz ładowania programów, może być wykorzystany do komunikacji z komputerem. Przykłady obsługi układu ACIA (**65C51N**):

```
uart_send:
  ; Sends a byte
  ;
  ; A - Data to be send

  sta ACIA_DATA_REG

  pha                           ; 3 cycles
  lda #WAIT_CYCLES_260us        ; 2 cycles
  jsr delay_a                   ; - cycles
  pla                           ; 4 cycles
  ;                             ; 9 cycles

  rts
```

```
uart_recv:
  ; Receives a byte
  ;
  ; RETURNS
  ; A - Data received

  lda ACIA_STATUS_REG
  and #ACIA_STATUS_RX_FULL
  beq uart_recv                  ; Loop waiting for character
```

```
lda ACIA_DATA_REG
rts
```

Powłoka Zaprogramowana w języku Assembler powłoka umożliwa podzielenie wywołanego polecenia na tokeny i uruchomienie podprogramu przypisanego danej komendzie.

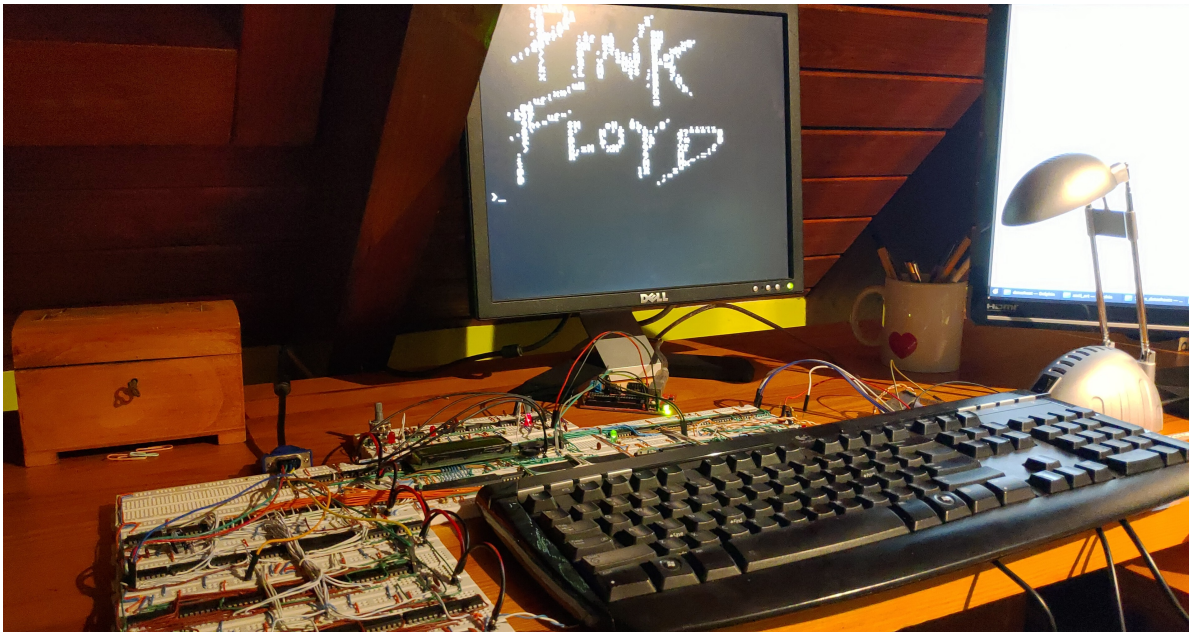
Wypisywanie Zaimplementowana została także funkcja z rodziny `printf` obsługująca formaty dla liczb całkowitych i własnej implementacji 16-bitowych liczb zmiennoprzecinkowych.

6.2 Mapa pamięci

Tabela 1: Mapa pamięci komputera

Lokalizacja	**Start**	**Koniec**	**Rozm.**	**Rozm.**	
RAM	#0000	#2FFF	$3 \cdot 2^{12}$	12288	bajtów
Aplikacja	#3000	#3FFF	2^{12}	4096	bajtów
UART	#6E00	#6EFF	2^{8}	256	bajtów
LCD	#6F00	#6FFF	2^{8}	256	bajtów
VRAM	#7000	#7FFF	2^{12}	4096	bajtów
ROM	#8000	#FFFF	2^{15}	32768	bajtów
			Suma	53760	bajtów
			Zostało	11776	bajtów

6.3 Przykłady działania



Rysunek 5: Komputer wraz z monitorem

Kod źródłowy Kompletny kod źródłowy wraz z plikiem `Makefile` znajduje się w folderze `asm`. Użyto assemblera `vasm6502_oldstyle`.

Filmy W folderze `filmy` znajdują się 3 filmy prezentujące działanie całego projektu. Prezentują one wyświetlanie grafik ASCII jak również inne komendy dostępne z poziomu powłoki. Jest również pokazane odbieranie kodów klawiszy i wyświetlanie ich na wyświetlaczu LCD.