

# LCD driver pro Digilab DIO2

## Dokumentace

### 1 Zadání

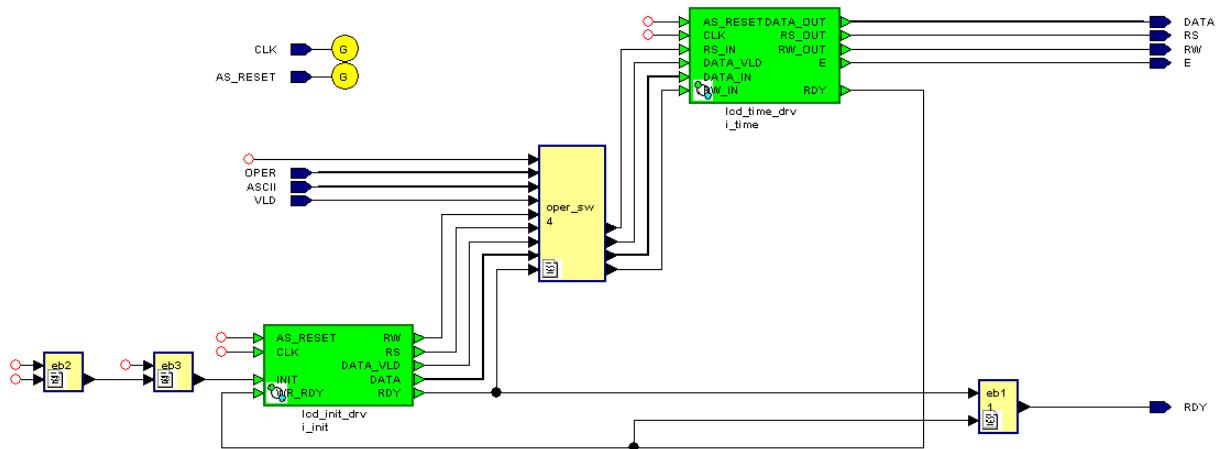
Cílem je udělat ovladač LCD displeje pro desku DIO2, tak aby maximálně zjednodušil použití.

### 2 Řešení - popis bloků

Řešení jsem rozdělil do 3 bloků (lcd\_time\_drv, lcd\_init\_drv a lcd).

#### 2.1 *lcd – top level*

Na následujícím schématu je zapojení top levelu.



Funkce je následující: Po pádu AS\_RESET do nuly je aktivní blok *lcd\_init\_drv*, jeho výstupy jsou přes MUX připojeny do *lcd\_time\_drv*. Dokončení inicializační sekvence LCD displeje (viz. Referenční manuál k DIO2) je signalizováno signálem *init\_rdy*, MUX je přepnuto na vstupní porty a může začít práce s displejem.

#### 2.1.1 Popis portů

```
entity lcd is
  Port (
    CLK          : in std_logic;
    AS_RESET     : in std_logic;
    -----
    OPER         : in std_logic_vector(1 downto 0);
    -- 00 - clear display
    -- 01 - return cursor home
    -- 10 - set DDRAM address
    -- 11 - write ASCII
```

```

ASCII      : in std_logic_vector(7 downto 0);
VLD       : in std_logic;
RDY       : out std_logic;
-- low level
DATA      : out std_logic_vector(7 downto 0);
RS        : out std_logic;
RW        : out std_logic;
E         : out std_logic);
end lcd;

```

Porty DATA, RS, RW, E jsou připojeny k LCD. Pro nás jsou důležité porty OPER, ASCII, VLD a RDY.

Bližší popis:

OPER – specifikace prováděné operace

00 = vymazání displeje, tj. vymazání DDRAM řadiče LCD

01 = Nastavení čítače adresy v řadiči na 0 + návrat kurzoru

10 = Nastavení pozice kurzoru pro zápis (adresa se zadává do ASCII, je použito spodních 7 bitů)

11 = Zápis ASCII na pozici kurzoru.

ASCII = ASCII kód znaku / pozice kurzoru

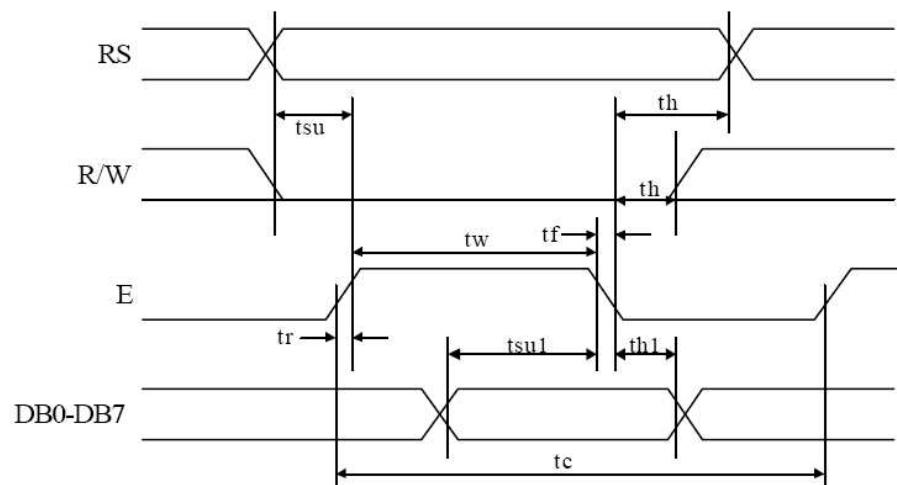
VLD – potvrzení vstupních signálů

RDY – připraven k další operaci

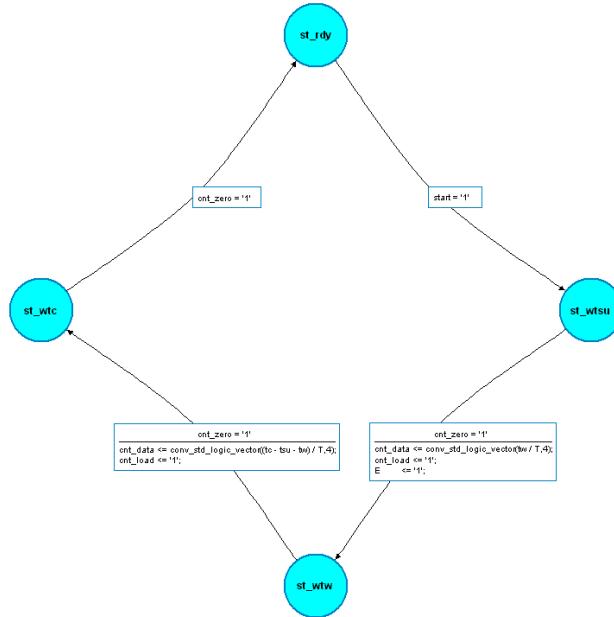
## 2.2 *Lcd\_time\_drv*

Tato komponenta se stará o správné časování signálů pro řadič displeje. Při přechodu signálu DATA\_VLD do 1, zaregistruje ostatní vstupní signály (tj. DATA\_IN, RW\_IN, RS\_IN) a spustí automat, který časuje signály podle následujících průběhů:

*LCD Write Cycle*

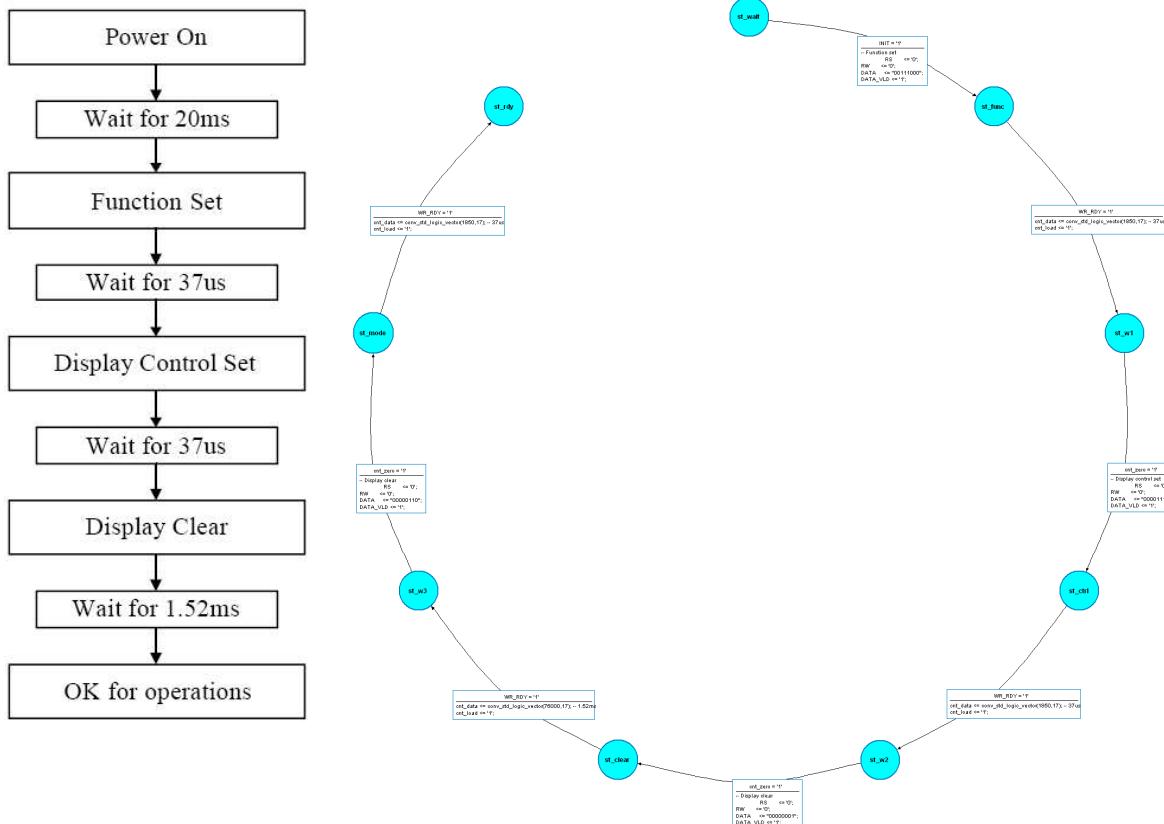


## Řídicí automat



### 2.3 lcd\_init\_drv

Tato komponenta se stará o inicializaci LCD. Podle referenčního manuálu je nutné před zápisem znaků provést následující sekvenci (vlevo). Na obrázku vpravo je stavový automat který toto zajišťuje.

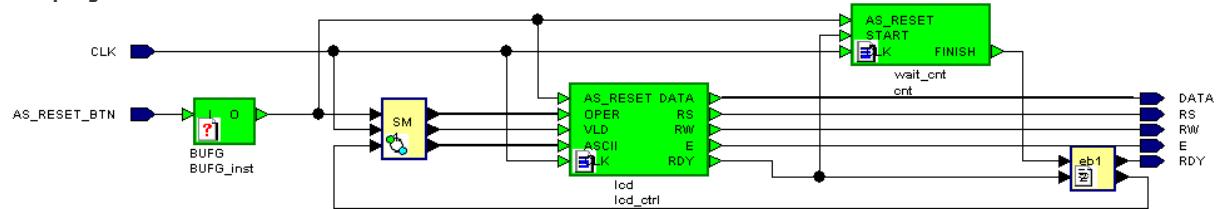


Aktivuje se signálem INIT a konec činnosti je indikován signálem RDY. V top\_level je INIT připojen na AS\_RESET.

### 3 Testování

Pro otestování funkce je možné použít připravený testbench, nebo přímo komponentu `Lcd_hw_test`, která po stisku tlačítka `AS_RESET` vypíše řetězec „ahoj“ na LCD.

Zapojení testbenče:



### 4 Známé problémy – náměty na vylepšení

#### 4.1 Problémy

Při testování v HW jsem zjistil, že po zápisu znaku je nutné cca 1ms počkat a pak je možné zapsat následující znak. V `Lcd_hw_test` jsem tento problém vyřešil pomocí čítače `wait_cnt`, který potřebné zpoždění zajistí.

#### 4.2 Možné vylepšení

##### 4.2.1 Přidání operace čtení BF

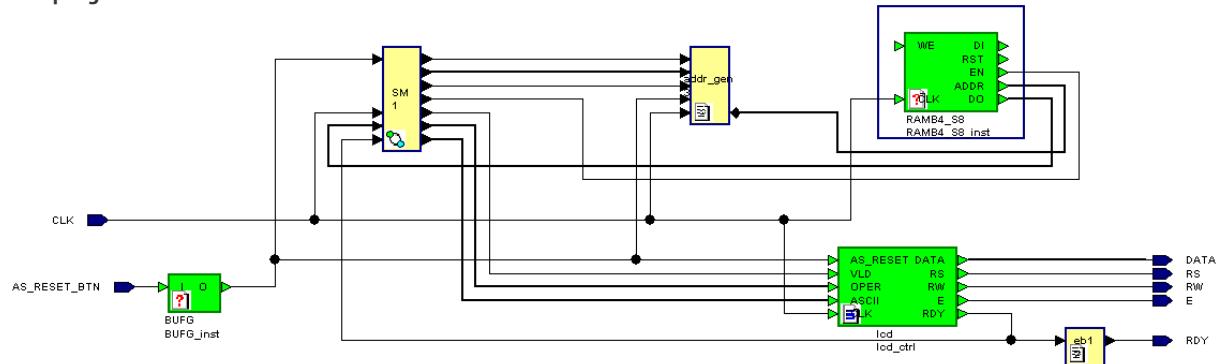
Rozšířit množinu operací o čtení příznaku zaneprázdnění (BF), nebo čist příznak automaticky v cyklu po každém zapsání do `Lcd_time_drv`. Tím by odpadlo použití čítače a výsledné řešení by bylo čistší. Je však nutné si uvědomit, že čtení se provádí přes datovou sběrnici a ta je společná pro LCD a CPLD přes které se přistupuje k dalším periferiím. Nadbytečné čtení by znemožňovalo rozumnou práci s CPLD.

##### 4.2.2 Zprovoznit `Lcd_hw_test2`

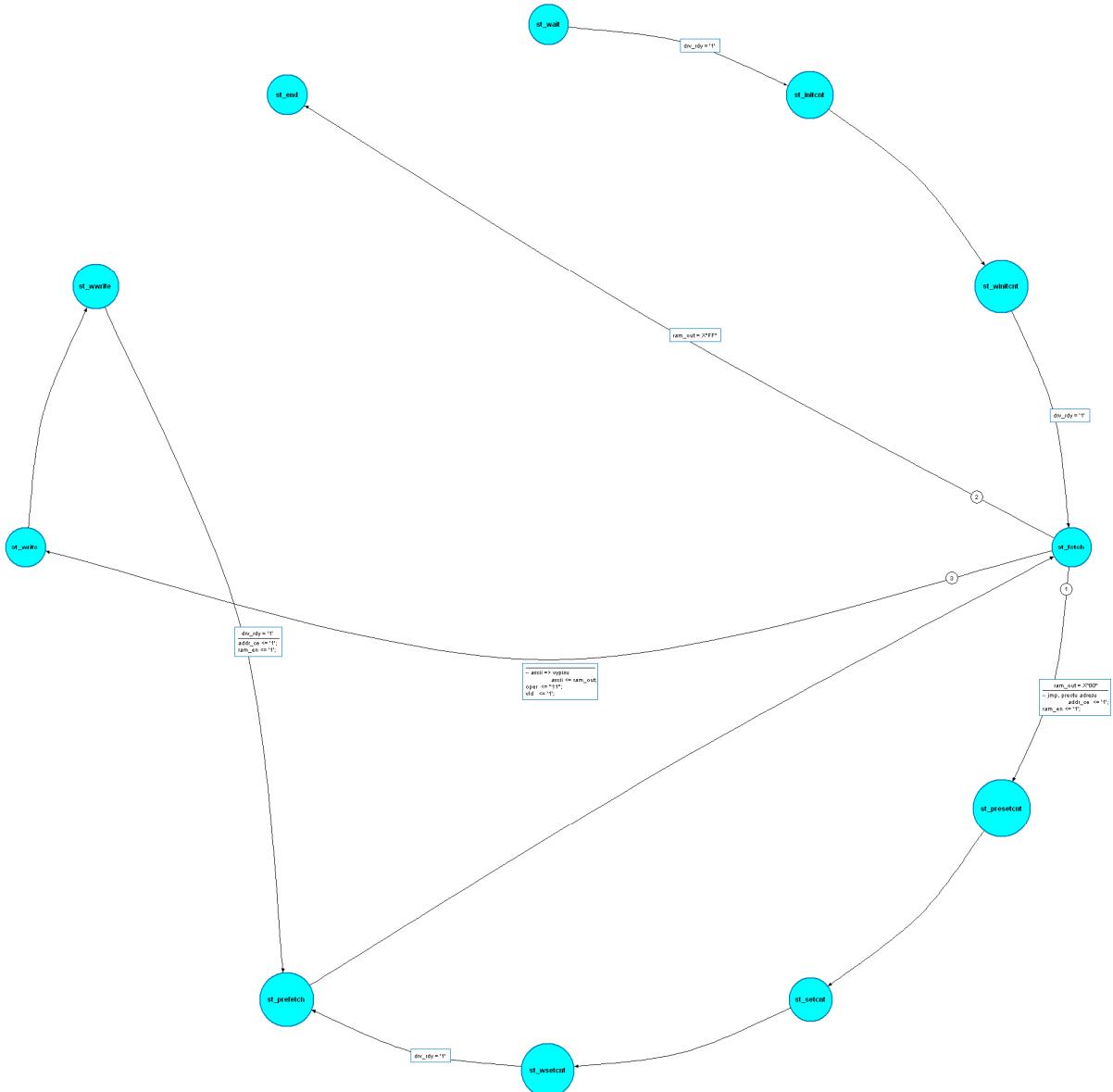
Jedná se o další test v HW. Pro uložení řetězce je použita BlockRAM a pro vyčítání automat. V automatu je implementována instrukce skoku (načtený Byte == 0, následující Byte == adresa kam skákat) a instrukce konce (načtený Byte == FF), vše ostatní je považováno za ASCII a vypsáno na LCD.

Je nutné tento test doplnit o zpoždění mezi jednotlivými zápisy do LCD, nebo o čtení BF. A celý test odladit v simulacích a HW.

Zapojení testbenče:



FSM:



## 5 Použitá literatura

- [1] Digilab DIO2 Reference Manual
- [2] Xilinx: Using Block SelectRAM+ Memory in Spartan-II FPGAs
- [3] Hazdra: Deska Spartan 2E