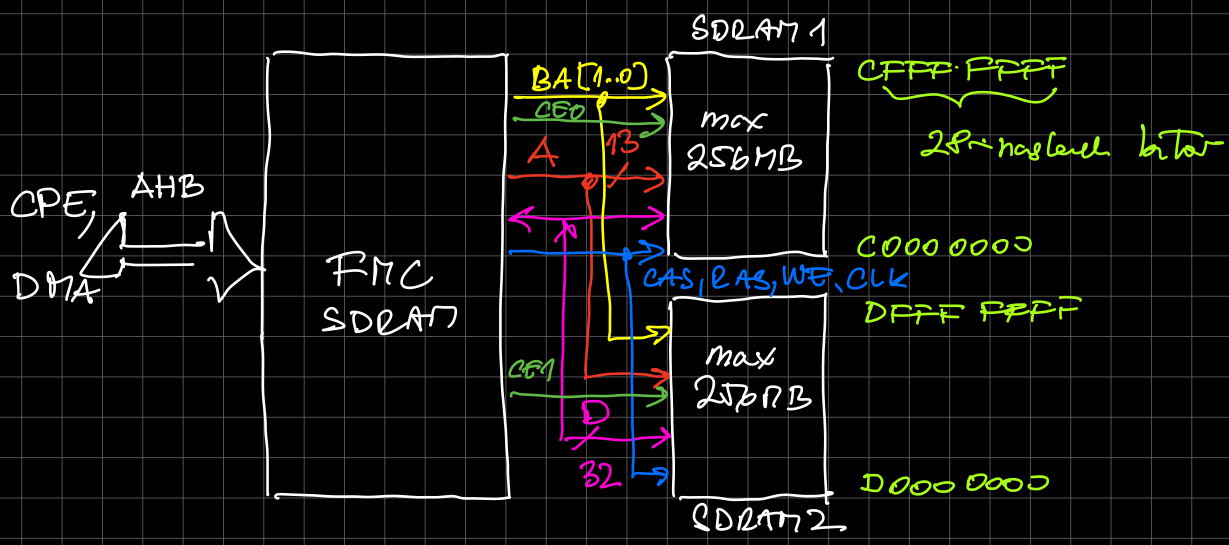
FMC + 128 Mbit SDRAM

Flexible Memory Controller

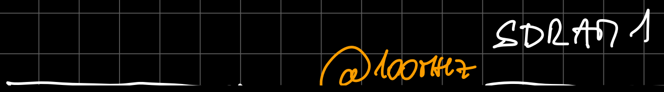
- NOR Flash
- NAND Flash
- SRAM
- SDRAM

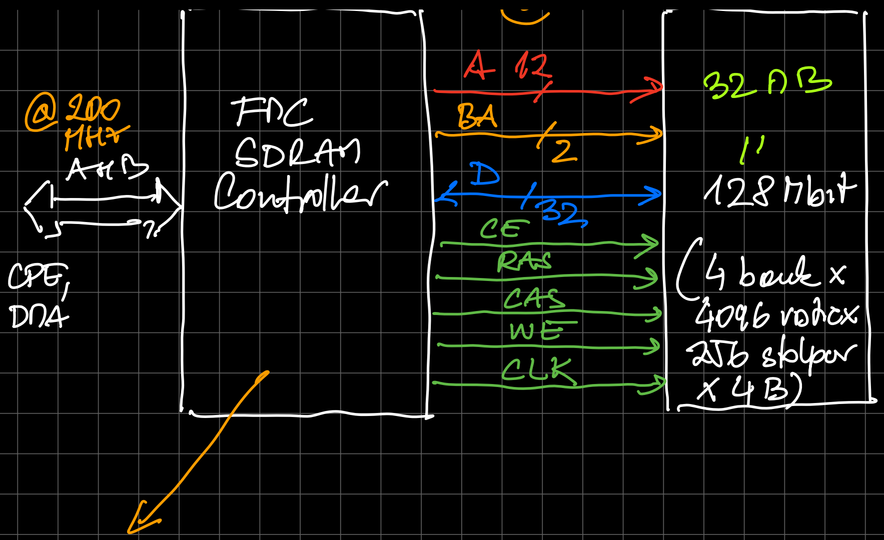
FMC SDRAM

→ lahko delo z dvema SDRAM čipoma



Mi imamo na voljo le en čip:





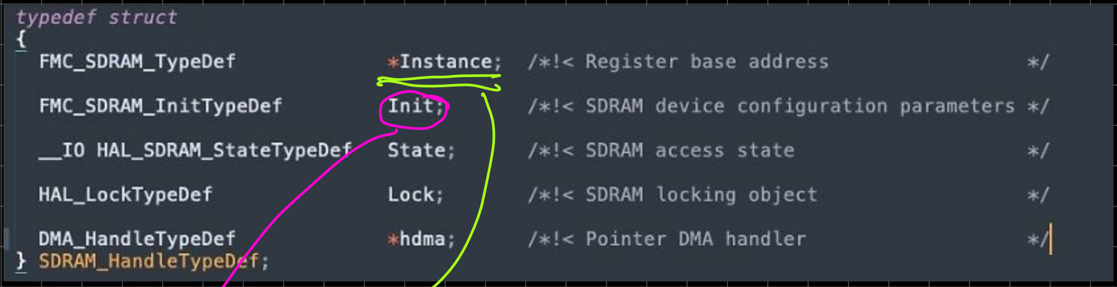
NASTAVLJIV:

- Število bank, ki naj jih naslavlja
- Število vrstic/stolpcev
- vsi potrebni časovni parametri
- pogostost osveževanja

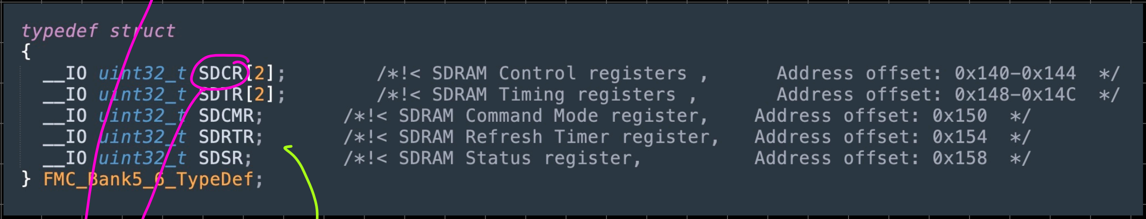
OMEJITVE:

- ne podpira eksplozijskih prenosov
  ampak lahko simuliramo
  - na en procesor LW ukaz, izvrši 4 READ ukaze proti SDRAM-u
    - 1. podatek vrne CPE, ostale 3 zapiše v svoj interni FIFO in potem če naslednji CPE dostopa te podatke, ga jih hitreje vrne

Za nastavljanje SDRAM krmilnika bomo uporabili HAL:



```
typedef struct
{
  FMC_SDRAM_TypeDef             *Instance;  /*!< Register base address                 */
  FMC_SDRAM_InitTypeDef         Init;       /*!< SDRAM device configuration parameters */
  __IO HAL_SDRAM_StateTypeDef   State;      /*!< SDRAM access state                    */
  HAL_LockTypeDef               Lock;       /*!< SDRAM locking object                  */
  DMA_HandleTypeDef             *hdma;      /*!< Pointer DMA handler                   */
} SDRAM_HandleTypeDef;
```

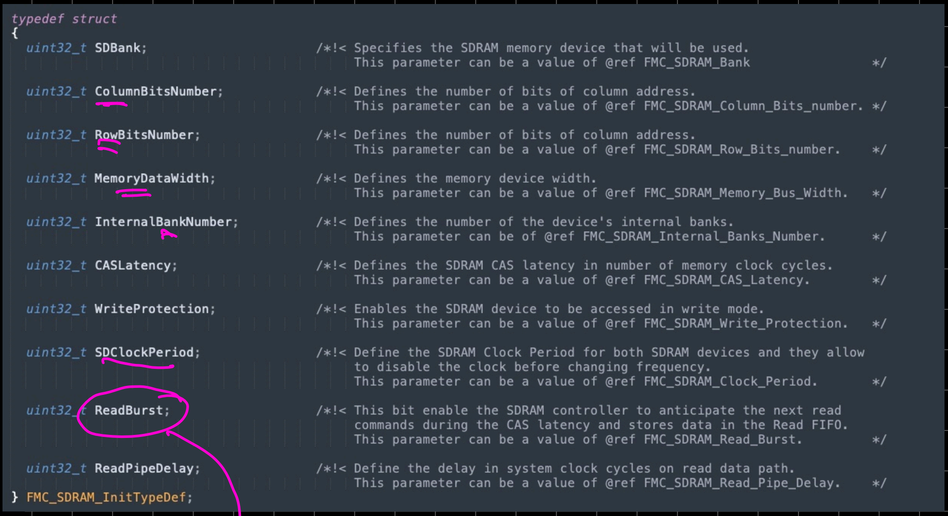


```
typedef struct
{
  __IO uint32_t SDCR[2];        /*!< SDRAM Control registers ,      Address offset: 0x140-0x144  */
  __IO uint32_t SDTR[2];        /*!< SDRAM Timing registers ,       Address offset: 0x148-0x14C  */
  __IO uint32_t SDCMR;       /*!< SDRAM Command Mode register,    Address offset: 0x150  */
  __IO uint32_t SDRTR;       /*!< SDRAM Refresh Timer register,   Address offset: 0x154  */
  __IO uint32_t SDSR;        /*!< SDRAM Status register,          Address offset: 0x158  */
} FMC_Bank5_6_TypeDef;
```

Celoten SDRAM krmilnik je z CPE viden kot le 7 registrov

Ker mi uporabljamo le en SDRAM čip, potrebujemo le 4 prve registre, zadnji je le statusni

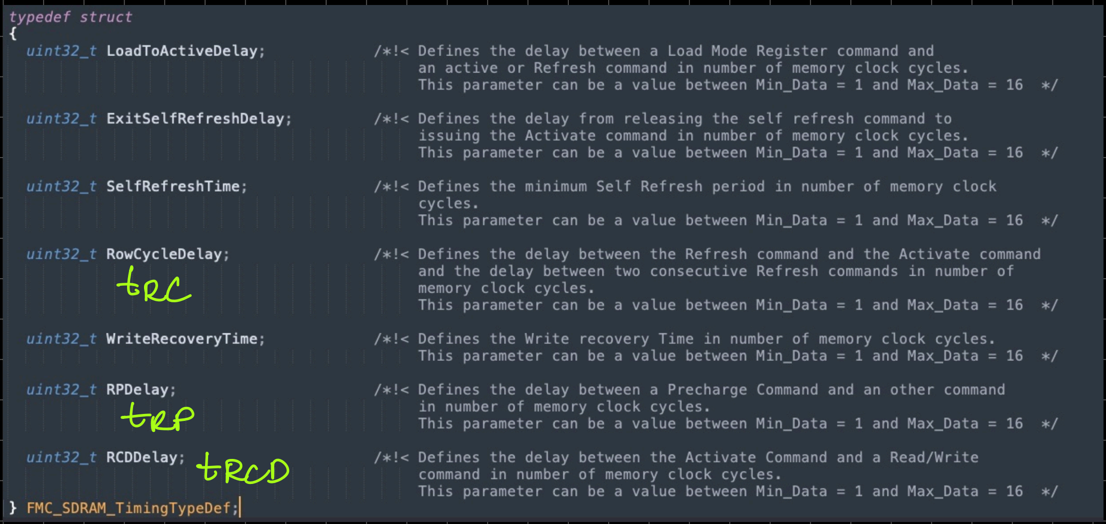
→ definira konfiguracijo za SDCR



```
typedef struct
{
  uint32_t SDBank;                      /*!< Specifies the SDRAM memory device that will be used.
                                             This parameter can be a value of @ref FMC_SDRAM_Bank              */

  uint32_t ColumnBitsNumber;            /*!< Defines the number of bits of column address.
                                             This parameter can be a value of @ref FMC_SDRAM_Column_Bits_number. */

  uint32_t RowBitsNumber;               /*!< Defines the number of bits of column address.
                                             This parameter can be a value of @ref FMC_SDRAM_Row_Bits_number.    */

  uint32_t MemoryDataWidth;             /*!< Defines the memory device width.
                                             This parameter can be a value of @ref FMC_SDRAM_Memory_Bus_Width.   */

  uint32_t InternalBankNumber;          /*!< Defines the number of the device's internal banks.
                                             This parameter can be of @ref FMC_SDRAM_Internal_Banks_Number.      */

  uint32_t CASLatency;                  /*!< Defines the SDRAM CAS latency in number of memory clock cycles.
                                             This parameter can be a value of @ref FMC_SDRAM_CAS_Latency.        */

  uint32_t WriteProtection;             /*!< Enables the SDRAM device to be accessed in write mode.
                                             This parameter can be a value of @ref FMC_SDRAM_Write_Protection.   */

  uint32_t SDClockPeriod;               /*!< Define the SDRAM Clock Period for both SDRAM devices and they allow
                                             to disable the clock before changing frequency.
                                             This parameter can be a value of @ref FMC_SDRAM_Clock_Period.       */

  uint32_t ReadBurst;                   /*!< This bit enable the SDRAM controller to anticipate the next read
                                             commands during the CAS latency and stores data in the Read FIFO.
                                             This parameter can be a value of @ref FMC_SDRAM_Read_Burst.         */

  uint32_t ReadPipeDelay;               /*!< Define the delay in system clock cycles on read data path.
                                             This parameter can be a value of @ref FMC_SDRAM_Read_Pipe_Delay.    */
} FMC_SDRAM_InitTypeDef;
```

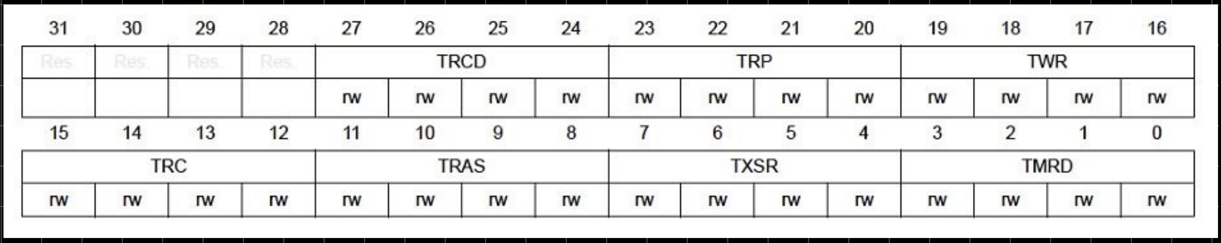
določa, ali se uporablja interni FIFO

Ta pod. struktura nam pomaga nastaviti bite v Timing registru → SDTR



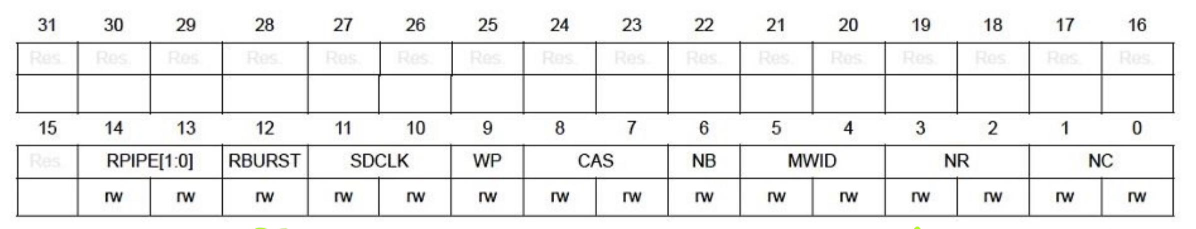
```
typedef struct
{
  uint32_t LoadToActiveDelay;          /*!< Defines the delay between a Load Mode Register command and
                                            an active or Refresh command in number of memory clock cycles.
                                            This parameter can be a value between Min_Data = 1 and Max_Data = 16  */

  uint32_t ExitSelfRefreshDelay;       /*!< Defines the delay from releasing the self refresh command to
                                            issuing the Activate command in number of memory clock cycles.
                                            This parameter can be a value between Min_Data = 1 and Max_Data = 16  */

  uint32_t SelfRefreshTime;            /*!< Defines the minimum Self Refresh period in number of memory clock
                                            cycles.
                                            This parameter can be a value between Min_Data = 1 and Max_Data = 16  */

  uint32_t RowCycleDelay;              /*!< Defines the delay between the Refresh command and the Activate command
           tRC                              and the delay between two consecutive Refresh commands in number of
                                            memory clock cycles.
                                            This parameter can be a value between Min_Data = 1 and Max_Data = 16  */

  uint32_t WriteRecoveryTime;          /*!< Defines the Write recovery Time in number of memory clock cycles.
                                            This parameter can be a value between Min_Data = 1 and Max_Data = 16  */

  uint32_t RPDelay;                    /*!< Defines the delay between a Precharge Command and an other command
           tRP                              in number of memory clock cycles.
                                            This parameter can be a value between Min_Data = 1 and Max_Data = 16  */

  uint32_t RCDDelay;  tRCD             /*!< Defines the delay between the Activate Command and a Read/Write
                                            command in number of memory clock cycles.
                                            This parameter can be a value between Min_Data = 1 and Max_Data = 16  */
} FMC_SDRAM_TimingTypeDef;
```

⇓



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	TRCD				TRP				TWR			
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

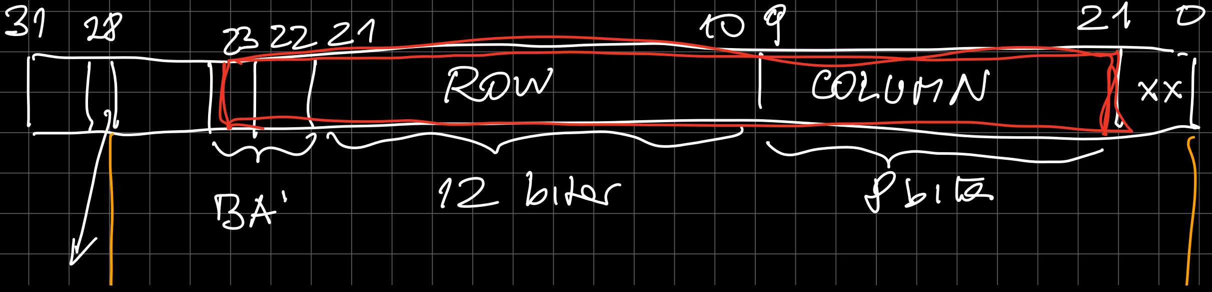
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRC				TRAS				TXSR				TMRD			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

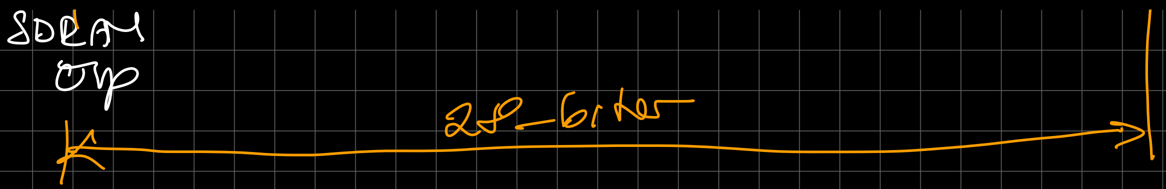


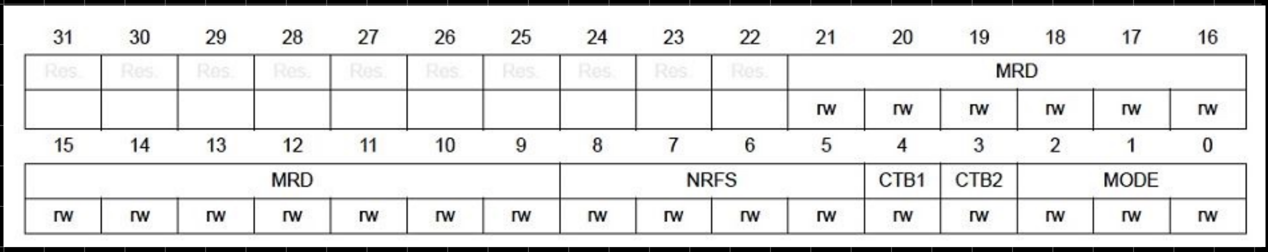
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	RPIPE[1:0]		RBURST	SDCLK		WP	CAS		NB	MWID		NR		NC	
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Preslikovanje naslova iz CPE/DMA (na AHB vodilu) v naslove proti SDRAM-u:







31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MRD					
										rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MRD							NRFS				CTB1	CTB2	MODE		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Osveževanje

Vsako SDRAM vrstico je treba osvežiti vsaj 1x v 64 ms

↓

SDRAM krmilnik osveži to tako, da v SDRAM pošlje ukaz AUTO REFRESH

⇓

SDRAM ima interni števec, ki naslavlja isto vrstico v vseh bankah in ob vsakem AR ukazu se števec poveča za 1

⇓

Imamo 4096 vrstic ⇒ v 64 ms moramo osvežiti vseh 4096 vrstic

za 1 vrstico 64ms/4096=15.625 μs

POGOSTOST OSVEŽEVANJA ⇐ to se vpiše v števec v FMC