# ST10 with MAC

## Direct Memory Access using MAC

**APPLICATION NOTE**

Direct Memory Access uses the ST10 MAC for address generation in external memory-external memory transfers. This application note supplies the machine code for external memory - ST10 internal memory transfers, and for external memory-external memory transfers made by Direct Memory Access using the MAC. For reference, the ST10 long addressing mode is summarized.

The ST10 instruction set contains special MAC instructions and two new addressing modes which supply the MAC with up to 2 new operands per instruction. The MAC contains a 16x16 multiplier, 40 bit accumulator a repeat unit and an address generator.

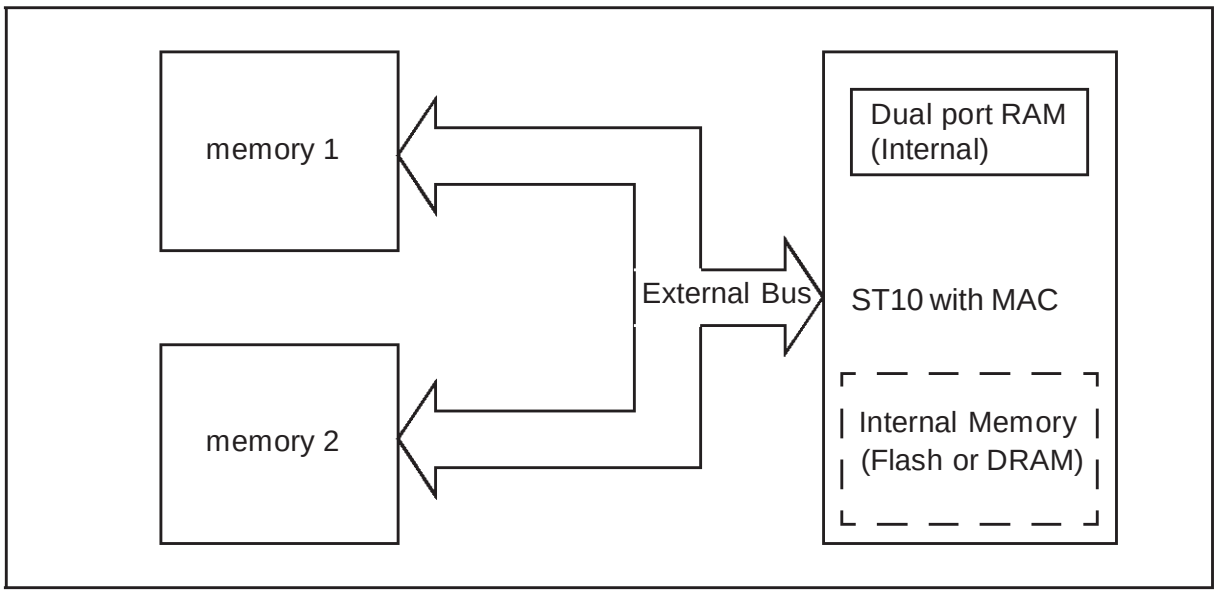


**Figure 1 System overview**

# 1 Memory transfer to ST10 internal memory

This machine code writes from external memory to internal memory, using the internal bus (stand-alone or super integrated).

It is not possible to execute code residing in the ST10 internal memory, while writing into it. The code below, must resides in another memory, e.g. external memory, X-memory, DPRAM.

```
mov    DPPX,    #addr_source      ; 10 bit prefix to point source memory
mov    DPPx,    #addr_dest        ; 10 bit prefix to point destination memory

mov    R0,      #addr_beg_source  ; 14 LSW of the address where to copy
mov    R1,      #addr_beg_dest    ; 14 LSW of the address destination
mov    R3,      #addr_end_source  ; 14 LSW of the last address to transfer
                                  ; Note that it can be done with a block size
                                  ; instead of a given address.
mov    R2,      [R0+]             ; Perform the copy from (R0) to R2
                                  ; Then increment the R0 register by 2.
mov    [R1+],   R2                ; Perform the copy from R2 to (R1).
                                  ; Then increment the R1register by 2.
cmp    R0,      R3                ; Compare current address (R0) to end address
                                  ; (R3). Loop until R0 >R3.
                                  ; Note that this test can be done for a given
                                  ; count.
jmpr   cc_ULE,  LOOP

RET                               ; Return, or RETI if done under interrupt
```

# 2 Memory transfer to external memory

This machine code writes from one external memory to another external memory, using Direct Memory Access with MAC address generation..

```
                mov     DPPX,    #XXXX   ; 10 bit prefix to point source memory
                mov     DPPx,    #XXXX   ; 10 bit prefix to point destination memory
                                         ; Solution if a 8 bit wide bus is used
                                         ; (3 lines below)
                EXTR    #2               ; Two following instructions used ESFR space
                mov     QX0,     #1      ; Store 1 in QX0 to allow byte transfers
                mov     QR0,     #1      ; Store 1 in QR0 to allow byte transfers

                mov     IDX0,    #dest   ; 14 LSW of the address where to copy mov
                        R0,      #source ;14 LSW of the address source
                mov     MRW,     #repeat ; use the maximum repeat depending on the
                                         ; timings allowed to perform copies
                                         ; Solution if a 16 bit wide bus is used
repeat MRW times CoMOV   [IDX0+], [R0+]  ; Perform the copy from (R0) to (IDX0)
                                         ; Then increment the two registers by 2.

                RET                      ; Return or RETI if done under interrupt
```

### DMA timing analysis

For a 50 MHz ST10 using a 16-bit wide demux bus:

- Register set-up takes 400 ns and only has to be done once.
- The first memory access takes 80ns. Each subsequent memory access takes 40 ns.

*Note* *This program only takes into account the program execution time and the time taken for memory accesses (including any waitstates on the source and the destination memory).*

# 3 Long addressing mode summary

The long addressing mode uses one of the four DPP registers to specify a physical 18-bit or 24-bit address.

Long 16-bit addresses are treated in two parts. Bits 0...13 specify the 14-bit data page offset, and bits 15...14 specify the Data Page Pointer (1 of 4). The lower ten bits of the selected DPP register are concatenated with the 14-bit data page offset to generate the physical 24-bit address (see figure below).

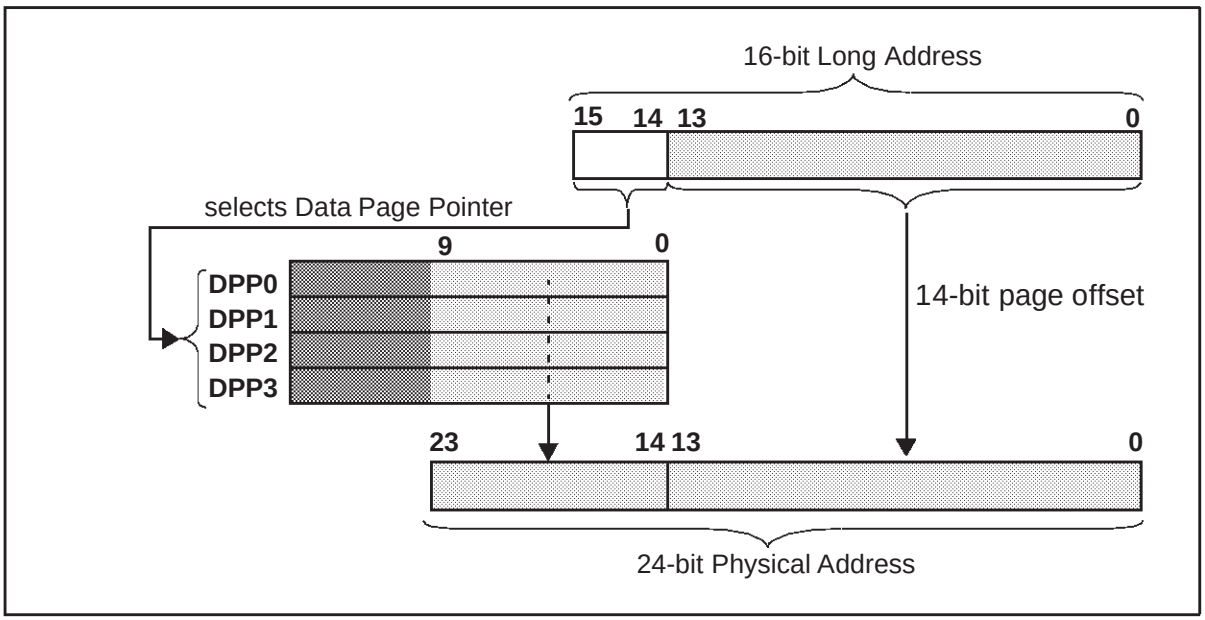


**Figure 2 Interpretation of a 16-bit long address**

The following address values are used: 00= DPP0, 01 = DPP1, 10 = DPP2, 11 = DPP3. Using the value 0x8C00 for addr_beg_source, the two MSB are 10. Therefore, DDP2 is used. If DPP2 = 0x0030, the resulting physical address is 0xC0C00.