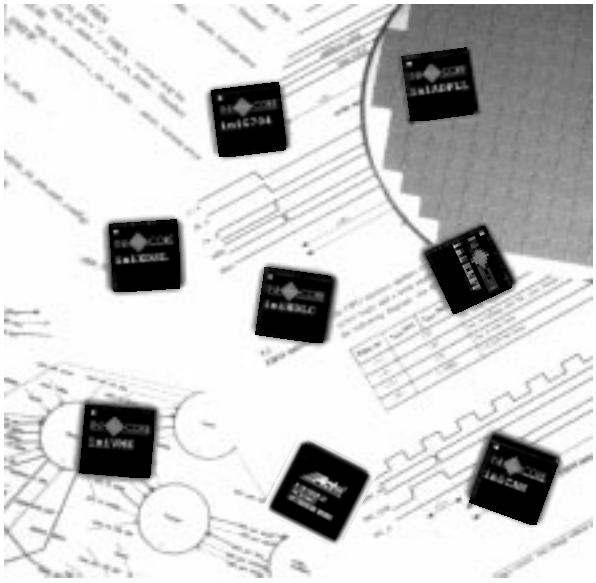
iniVME

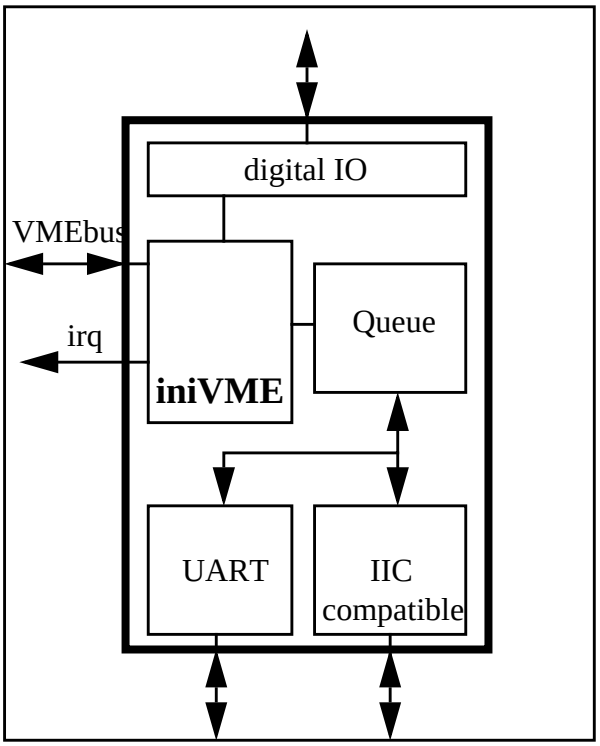
data sheet



Features:

- Flexible slave VME controller
- 16bit data, 24bit address
- Full interrupt controller (ROAK)
- Control signals for external drivers and drivers on chip
- Synchronous user side interface for registers, peripherals and memory
- User definable waitstates
- Synchronous, reliable design
- Expandible to full set of VME features
- Silicon proven design
- Suitable for ASIC and FPGA
- Samples available

Typical application:



INICORE - the reliable Core and System Provider. We provide high quality IP, design expertise and leading edge silicon to the industry.

VME, developed for industrial computing and controlling systems, is a global accepted standard bus, mostly used with Motorola CPUs.

Although a large variety of peripheral cards are available, there is a need for designing your own subsystem, based on VME. For that purpose, INICORE created the structured VHDL VME model for simulation and synthesis on any target technology.

INICORE's strategy means not competing with standard chip manufacturers, but using ASIC technologies for 'system on chip' design, where existing bus structures like VME must be combined with highly specific data processing and functionality. iniVME with its flexible interfaces is easy to combine with special IO, peripherals etc. for high performance applications as well as for low cost.

INICORE is owner of application proved VME cores and is able to provide turn key solutions for your special applications.
Thanks to our advanced methodology and design style, we manage fast prototyping and low volume production perfectly on FPGAs, with an open migration path to Gate Array technologies.

US Sales Office:
INICORE INC.
5600 Mowry School Road, Suite 180,
Newark, CA 94560
Tel: 510 445 1529 Fax: 510 656 0995
E-mail: ask_us@inicore.com
Web: www.inicore.com

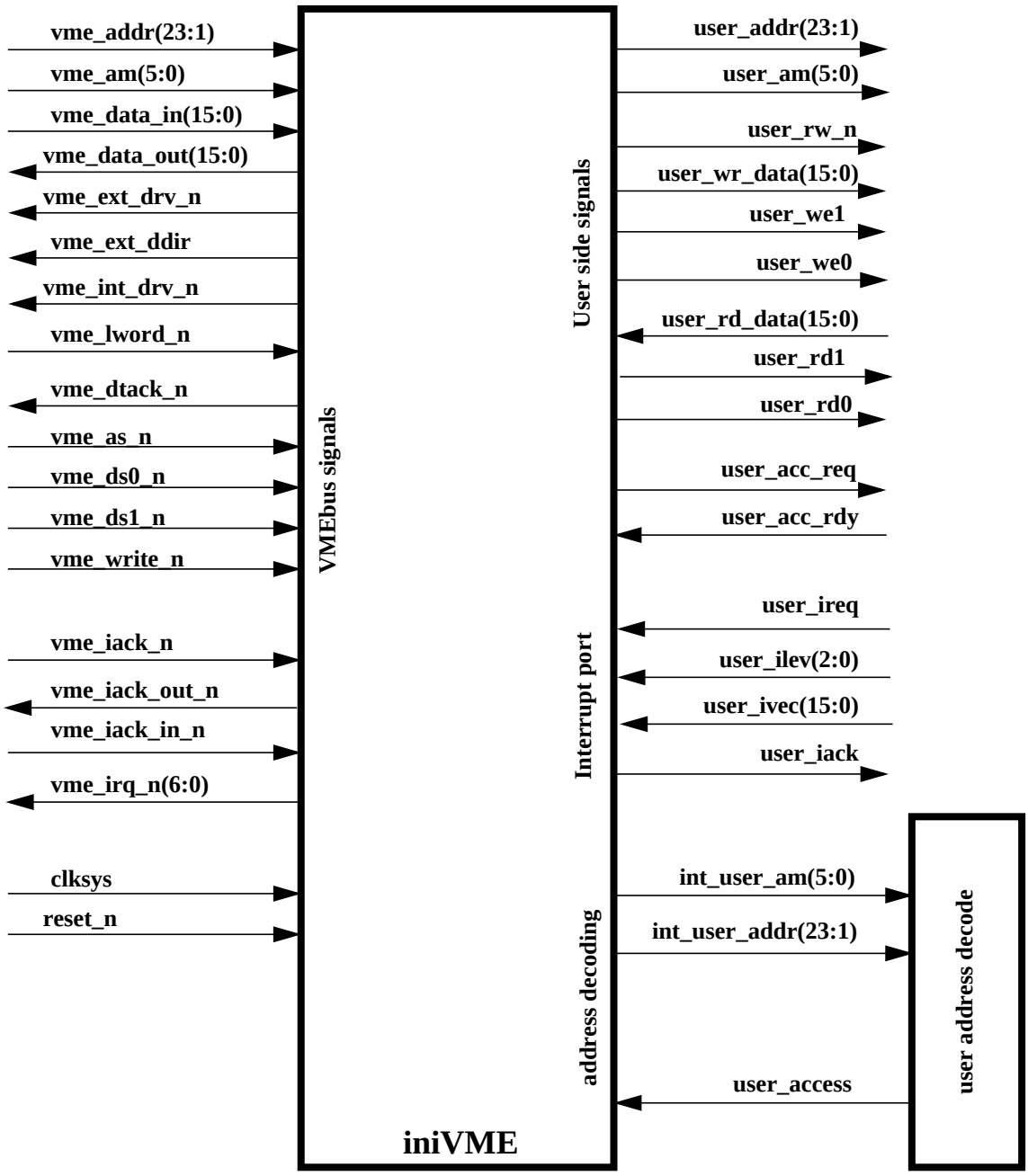
INICORE AG
Mattenstrasse 6a,CH-2555 Brügg, Switzerland
Tel: ++41 32 374 32 00, Fax: ++41 32 374 32 01
E-mail: ask_us@inicore.ch
Web: www.inicore.ch

Ref.Nr.: 361-DS-11 / 08/99

1 Overview

The iniVME slave core provides an easy to use, synchronous parallel interface towards custom logic (called user side), including full interrupt support. VME compatible external drivers can be directly connected to the iniVME. Examples are shown in the application section.

The following pictures shows all inputs and outputs:



2 IO description

The following part lists the input and output ports of the iniVME core and provides an overview of their functionality.

2.1 General inputs

These pins are used to clock and initialize the whole iniVME core. There is just one other clock than the clk signal. The vme_as_n registers the vme_addr and the vme_am on falling edges.

pin name	type	description
clk	in	system clock
reset_n	in	asynchronous system reset, active low

2.2 VME Bus

These pins are used to control data transfer through the VME interface.

pin name	type	description
vme_addr[23:1]	in	VME address bus input
vme_am[4:0]	in	VME address modifier input
vme_data_in[15:0]	in	VME data bus input (after bus driver)
vme_data_out[15:0]	out	VME data bus output (goes to bus driver)
vme_ext_drv_n	in	Active low drive enable signal for external bidirectional data bus drivers.
vme_int_drv_n	in	Active low drive enable signal for internal bidirectional data bus drivers.
vme_ext_ddir	in	Direction control signal for external bidirectional data bus drivers: ‘1’ to VME bus ‘0’ from VME bus
vme_lword_n	in	VME long word access indicator, low active
vme_dtack_n	out	Data transfer acknowledge output, active low. Has to be connected to open collector driver.
vme_as_n	in	VME address strobe: clocks with falling edge the internal synchronisation signals like vme_addr and vme_am. vme_as_n is also used as data signal for access start detection.
vme_ds0_n	in	Data strobes 0, active low
vme_ds1_n	in	Data strobes 1, active low
vme_write_n	in	Read/write signal, active low
vme_iack_n	in	Interrupt acknowledge, active low
vme_iack_in_n	in	Interrupt acknowledge daisy chain, active low
vme_iack_out_n	out	Interrupt acknowledge daisy chain, active low
vme_irq_n[6:0]	out	Interrupt, active low. Have to be connected to open collector driver.

2.3 User Side Interface

This part describes the user side interface, which is designed for easy register and memory access.

2.3.1 Signal description

The entire user side interface is part of the 40Mhz clock domain. The application dependent address modifier logic is not integrated in the core. This allows the user to build his own address decoding logic without changing the code of the iniVME. For convenience, more signals are provided than a minimal approach would require. This simplifies the design of the user side logic. Timing information can be found in paragraph *3 Timing.*

pin name	type	description
int_user_addr (23:1)	out	Registered VME address bus (by falling edge of vme_as_n)
int_user_am (5:0)	out	Registered VME address bus modifier (by falling edge of vme_as_n)
user_access	in	User access signal. The user has 50ns time to decode the address and asserting the user_access signal when addressed.
user_acc_req	out	User access request: Active high until user_acc_rdy acknowledges the request (or VME bus error occurs)
user_acc_rdy	in	User side acknowledgement signal. Active one event which finishes user side access.
user_addr(23:1)	out	Registered VME address bus
user_am(5:0)	out	Registered VME address bus modifier
user_wr_data (15:0)	out	Write data bus. MSB Data (15:8) is valid while user_we1='1' LSB Data (7:0) is valid while user_we0='1'
user_rd_data (15:0)	in	Read data bus. Must be valid when user_acc_rdy is 1.
user_rw_n	out	Data read/write_not signal. '1' : read data '0' : write data
user_we1	out	MSB Data write enable signal. While active one, user_wr_data(15:8) is valid. Returns to zero when user side acknowledgement with user_acc_rdy happened.
user_we0	out	LSB Data write enable signal. While active one, user_wr_data(7:0) is valid. Returns to zero when user side acknowledgement with user_acc_rdy happened.
user_rd1	out	User read MSB enable. An active one indicates VME read request. Returns to zero when user side acknowledgement with user_acc_rdy happened.
user_rd0	out	User read LSB enable. An active one indicates VME read request. Returns to zero when user side acknowledgement with user_acc_rdy happened.
user_ireq	in	Interrupt request. Active 1 indicates that an interrupt is pending and a VME interrupt will be generated. Must return to zero with user_iack = 1.

pin name	type	description
user_iack	out	Interrupt acknowledgement. An active one event indicates the end of a valid interrupt acknowledge cycle.
user_ilev(2:0)	in	Interrupt level
user_ivec(15:0)	in	Interrupt vector

2.3.2 User address decoding

To decide if an access is for the actual card or not, a decode logic has to be added externally. The signals int_user_addr and int_user_am allow a standard memory mapped address decoding scheme. The address modifiers may be used to differentiate between user / system access, io or memory etc. It is not mandatory to use the address modifiers.

The int_user_addr and int_user_am signals are latched with the falling edge of vme_as_n and are immediately valid. The user address decode logic has to be designed for a delay time shorter than 50ns (2 cycle path). This gives more flexibility for the user and lowers the constraints for this part of logic.

3 Timing

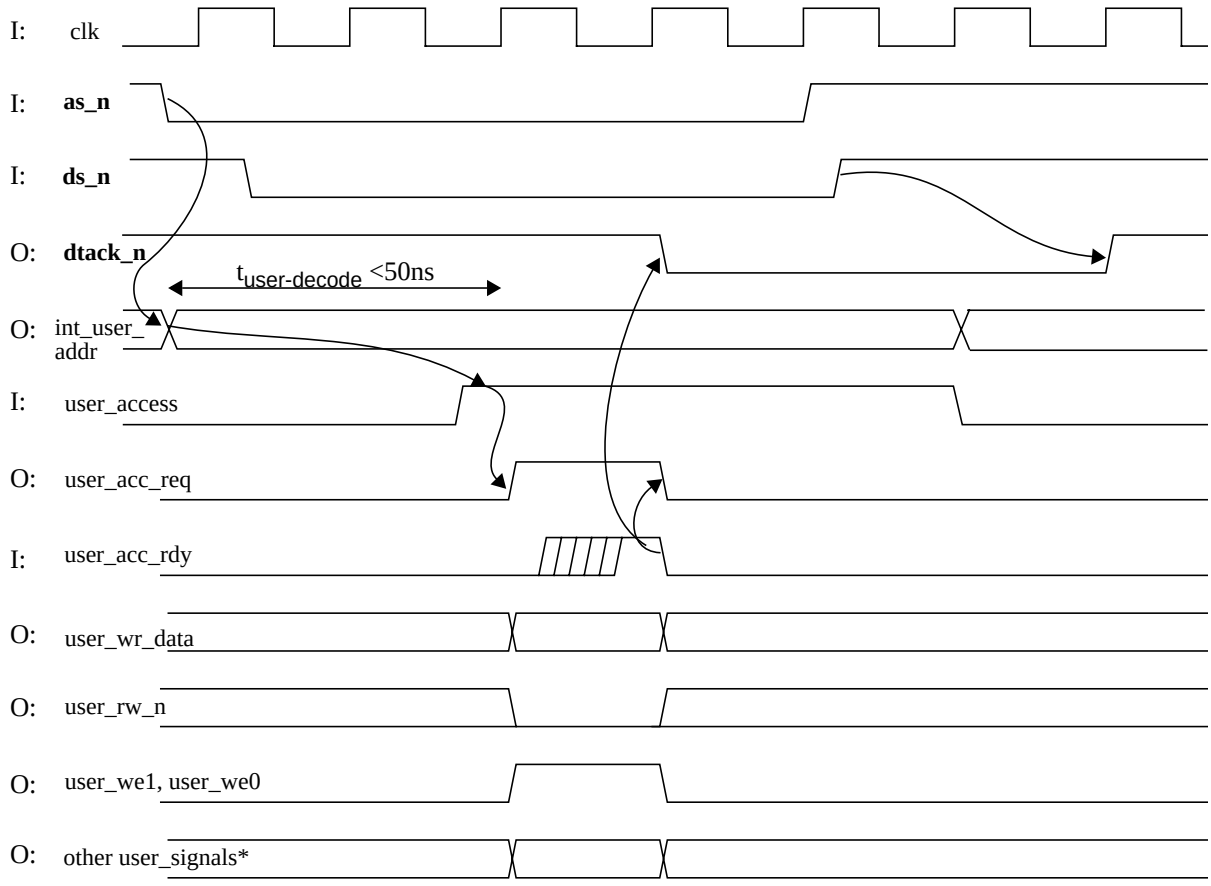
This part describes the timing of the iniVME solution.

3.1 General Information

All VMEbus inputs are synchronized to the local clock domain. A frequency of 40MHz is required to detect all level changes on the VME bus correctly.
Outputs are all synchronous to the iniVME clock domain. Only int_user_addr and int_user_am are registered by the falling edge of vme_as_n signal.

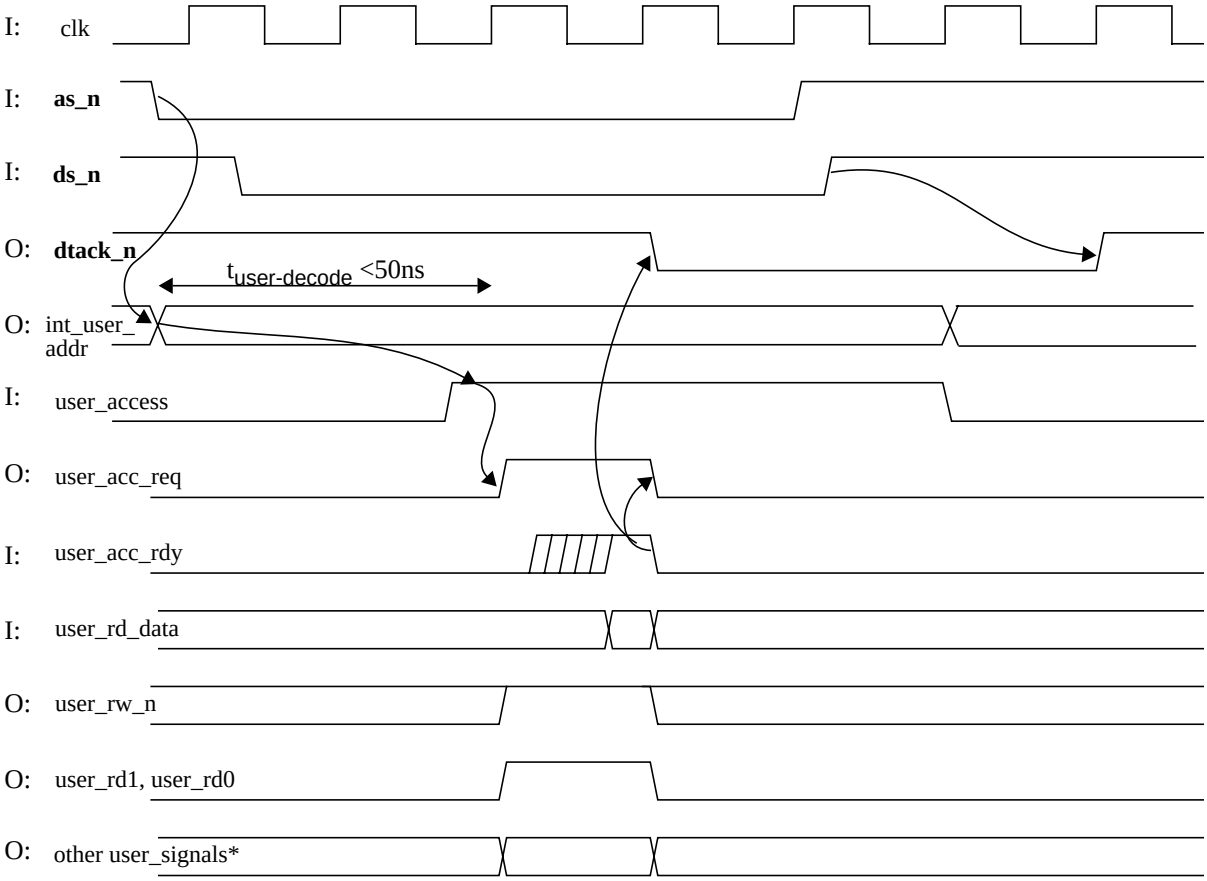
3.2 User access timing

3.2.1 Write access, no user waitstates



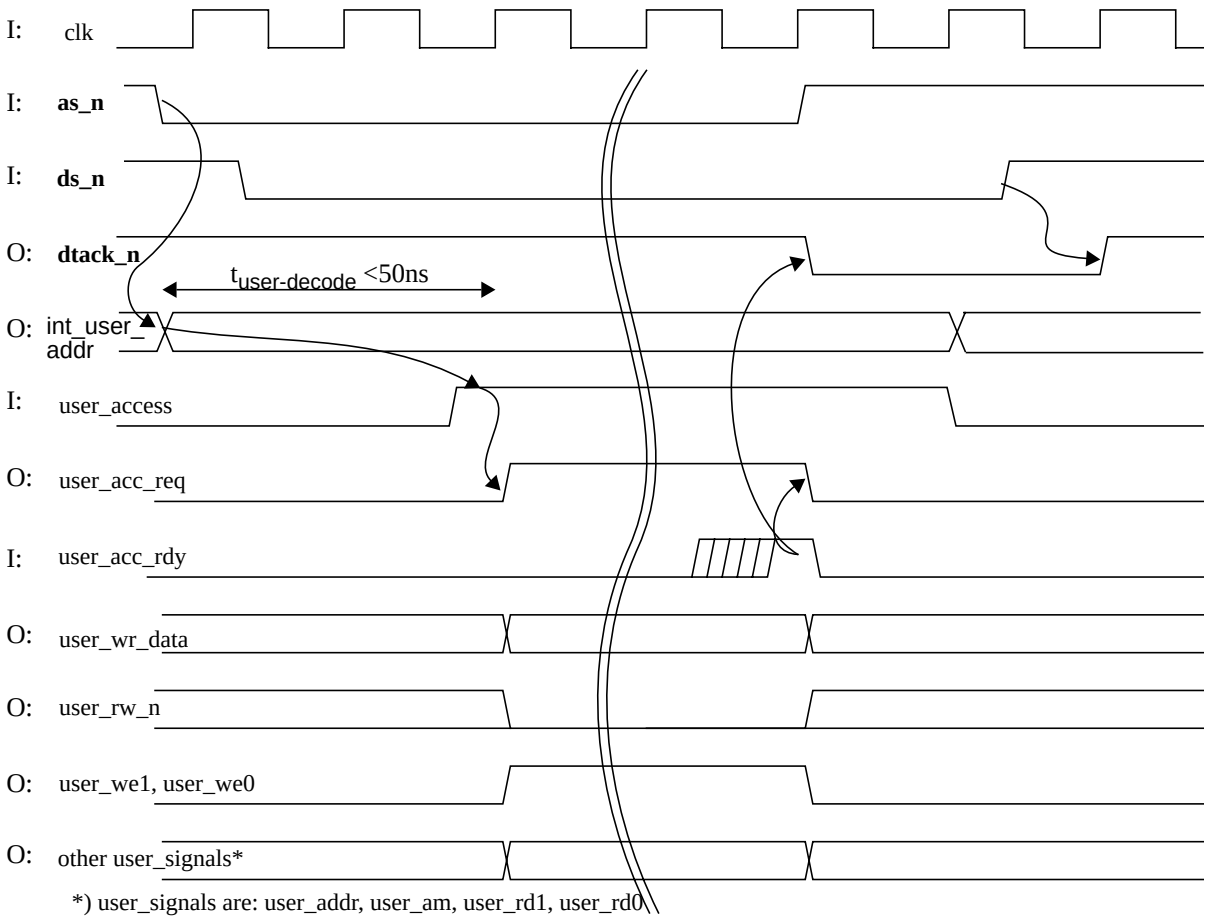
*) user_signals are: user_addr, user_am, user_rd1, user_rd0

3.2.2 Read access, no user waitstates



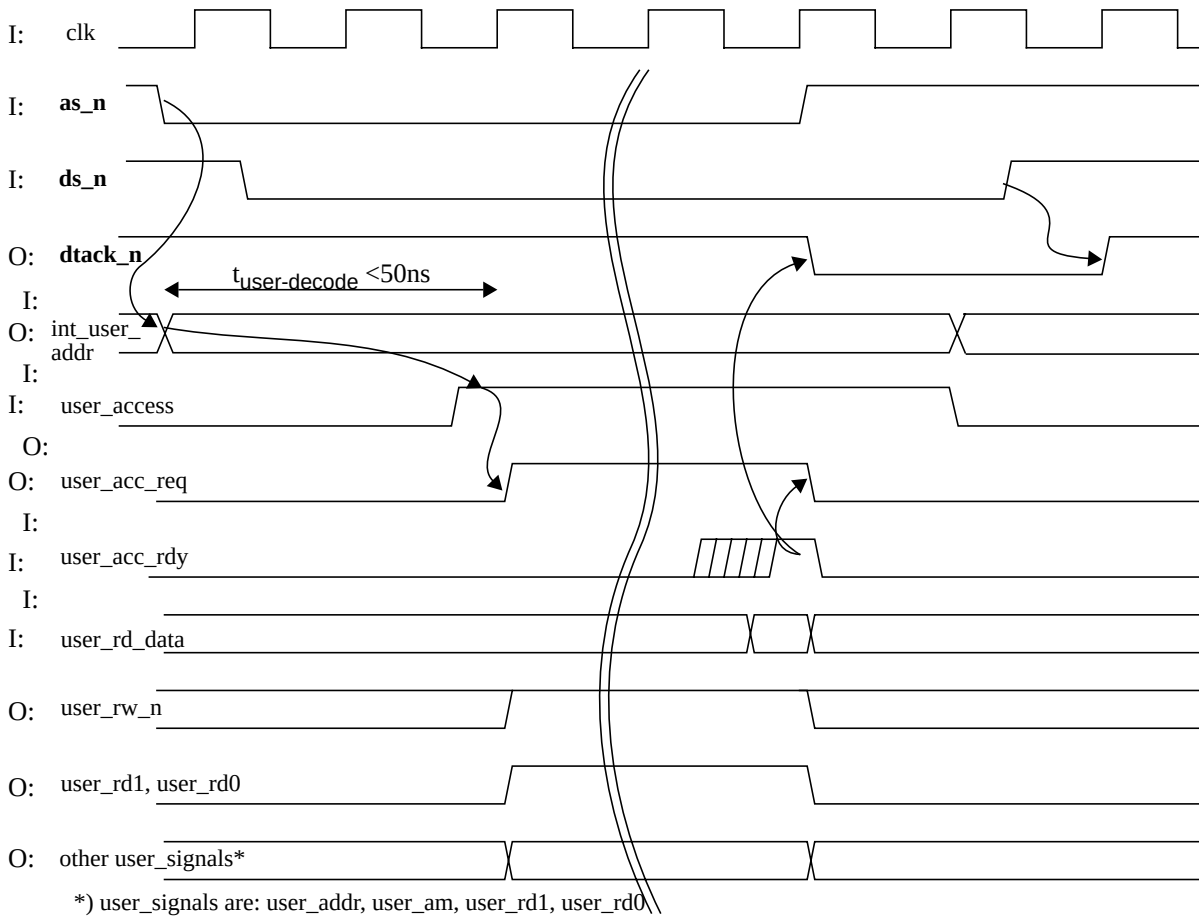
*) user_signals are: user_addr, user_am, user_we1, user_we0

3.2.3 Write access, user waitstates



*) user_signals are: user_addr, user_am, user_rd1, user_rd0

3.2.4 Read access, user waitstates



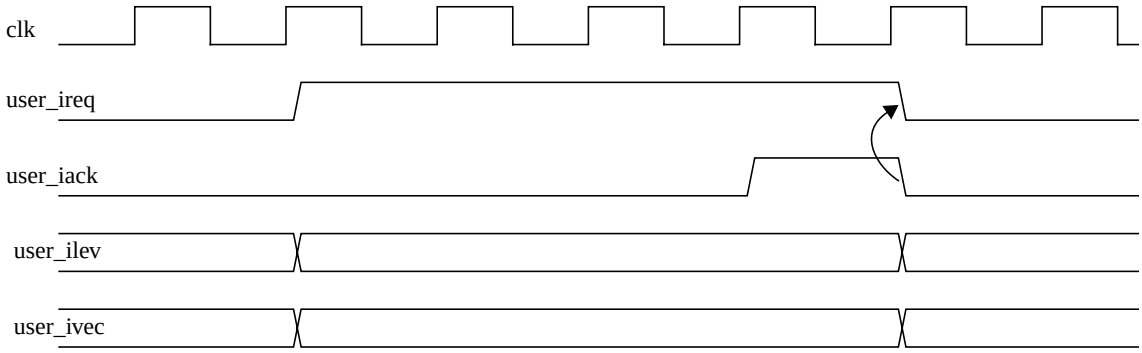
*) user_signals are: user_addr, user_am, user_rd1, user_rd0

3.2.5 Interrupt Timing

Interrupt requests to the VME bus are signalled by the active high user_ireq. Depending on the user_ilev (interrupt level), the vme_irq_n(x) will be asserted. As soon as the interrupt is acknowledged by the VME bus, the user_iack event is asserted which will deassert user_ireq on the user side.
The interrupt type is ROAK (Release On AcKnowledge), which means that the interrupt request on the VME bus is release immediately at the interrupt acknowledge cycle.

The interrupt vector is 16bits, so a D08 and D16 interrupter can be implemented. Please note that for a D08 (only bits (7:0) are used by the interrupt handler), the bits (15:8) should be programmed as 0xFF.

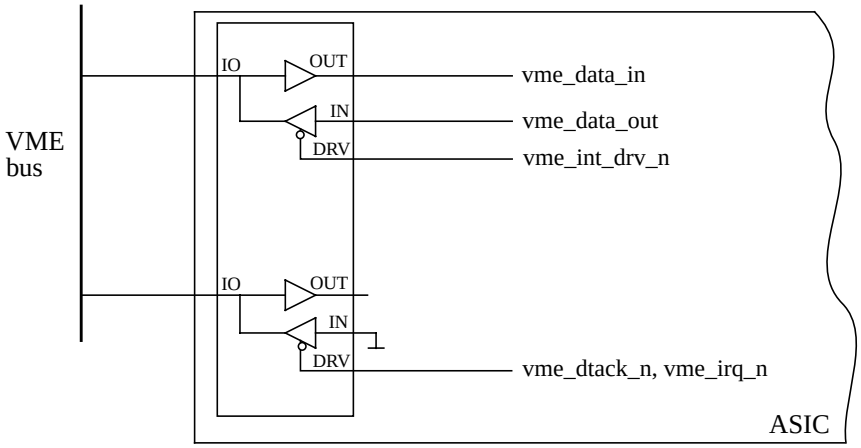
user_ilev and user_ivec have to be stable for the whole time period where user_ireq is high.



4 Applications

The VME specification requires 64 mA bus drivers. The bus drivers are not integrated in the iniVME core. Since the FPGA are not able to drive this load, external drivers might be used. The driver should have the following functions:

This diagram shows how internal buffers can be used to directly interface with the VME bus. These drivers have to fulfil VME specifications.



This diagram shows how external standard parts can be used to interface with the VME bus. The used 74xx parts have to fulfil VME specifications. Please note that outputs like vme_dtack_n, vme_irq_n etc. are push-pull outputs and require and external open collector driver like the 74641.