It's no trick...
it's a vision system

Vision
Components

The Smart Camera People

Encoder Interface

Application Notes

Revision 1.1 July 2007
Document name: Encoder.pdf

Foreword and Disclaimer

This documentation has been prepared with most possible care. However Vision Components GmbH does not take any liability for possible errors. In the interest of progress, Vision Components GmbH reserves the right to perform technical changes without further notice.
Please notify **support@vision-components.com** if you become aware of any errors in this manual or if a certain topic requires more detailed documentation.
This manual is intended for information of Vision Component's customers only. Any publication of this document or parts thereof requires written permission by Vision Components GmbH.

Trademarks

Code Composer Studio and TMS320C6000, Windows XP, Total Commander, Tera Term, Motorola are registered Trademarks. All trademarks are the property of their respective owners.

References

Since the VC4XXX smart camera family employs a TI processor, the programming environment and functions for the VC20XX cameras can be used for this camera.

Further References under "Support + Download" on www.vision-components.com**:**

„Support News" – for up to date information on VC Software and Documentation.

„Knowledge Base / FAQ" - searchable Database with latest software developments, frequently asked questions and demo programs.

"Download Areas" for all documentation and Software downloads – refer to the following table:

Description	Title on Website	Download Area
Schnellstart VC 🇩🇪.	Schnellstart VC Smart Kameras	Registered User Area ▸Getting Started VC SDK TI
Introduction to VC Smart Camera programming	Programming Tutorial for VC20XX and VC40XX Cameras	Registered User Area ▸Getting Started VC SDK TI
Demo programs and sample code used in the Programming Tutorial	Tutorial_Code	Registered User Area ▸Getting Started VC SDK TI
VC4XXX Hardware Manual	VC4XXX Smart Cameras Hardware Documentation	Public Download Area ▸Hardware Documentation VC Smart Cameras
VCRT Operation System Functions Manual	VCRT 5.0 Software Manual	Registered User Area ▸Software documentation VC Smart Cameras
VCRT Operation System TCP/IP Functions Manual	VCRT 5.0 TCP/IP Manual	Registered User Area ▸Software documentation VC Smart Cameras
VCLIB 2.0 /3.0 Image Processing Library Manual	VCLIB 2.0/ 3.0 Software Manual	Registered User Area ▸Software documentation VC Smart Cameras

The Light bulb highlights hints and ideas that may be helpful for a development.

This warning sign alerts of possible pitfalls to avoid. Please pay careful attention to sections marked with this sign.

Author: Peter Neuhaus, VC Support, **mailto:support@vision-comp.com**

Table of Contents

1 Introduction and FAQ's

Which models include an incremental encoder interface?

- All VC4XXX "Standard - " and "High End Range" Smart Cameras (VC4038 to VC4068 and VC4438 to VC4472) – not the "Entry Level Range" VC4018 and VC4016 or the VCSBC4018 and VCSBC4016 board cameras.

What applications can be done with the encoder interface?

- Highly accurate synchronization of the image acquisition with
  - rotating machinery (e.g. indexing tables, rotating shafts, robot axis)
  - and linear motion (linear axis, conveyor belt, etc.
- Additional features (under development):
  → Tracking of inspected part from entering an inspection system (e.g. light barrier) to image acquisition and sorting in good / bad parts in hardware (i.e. interrupt controlled).

How does it work?

- Like VC's standard interrupt controlled hardware trigger (see trigin.c demo program).

Differences to "normal" hardware trigger mode:

Hardware trigger signal counter (24 bit – 25 MHz):

→ Images are not taken at every trigger input, but only every X number of signals. A fast hardware counter counts the trigger input signals and the next image is taken at the counter value set in advance (a counter "Reload Value" sets the counter value for the image after the next one in order to manage short image timing intervals).

Direction detection and Zero pulse:

→ Automatic direction detection evaluating the A and B encoder signals, avoiding false motion detection by bouncing encoder signals. Inverted Signals (5 channel mode) are not supported.

→ Synchronization with rotating shaft angle / linear position by resetting the counter using the Zero- (N-) pulse of the encoder (either only during the first Zero pulse detection or at every Zero pulse).

→ The additional serial RS232 interface can not be used at the same time as the additional encoder input lines (here B+ and 0+) as these input lines are now used for these additional channels as shown.

→ The trigger counter can also be used with the standard trigger input only (single line mode). Direction detection or Zero pulse synchronization is not supported in this case. In this case setting the camera to encoder mode is not required. The use of one encoder signal (for instance A+) can be used together with the additional serial interface.

1.1 Suitable Encoder to use with the VC4XXX encoder interface:

→ All 3 channel (A, B, Zero or N) encoder are supported that meet the electrical specifications of the TTL trigger input as detailed in the VC4XXX manual.

Special Advice:

→ Using the 5V out (pin 2, brown cable) of the trigger interface as encoder power supply is possible if:

- The encoder power supply is rated 5V, max 50mA
- And the encoder output signal high match the TTL level trigger input (2.4V to 5V). Use short or additionally shielded cables with low input signal voltage!

→ Using a “**push- pull**” TTL encoder avoids using pull down or pull up resistors at the camera encoder inputs.

This way the incremental encoder can be connected directly to the trigger interface, just using the trigger cable without any additional wiring required.

See the following specifications of an incremental encoder type 2400 from Fritz Kübler GmbH (**www.kuebler.com**):

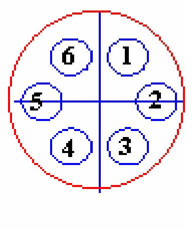
- Encoder power supply: UB = 5 to 24V, max. current drawn: 50 mA
- Signal high = UB - 2.5 V, Signal low = max 0.5 V
- Output circuit: Push – Pull
- Pulses per revolution: 4 to 1080

- The high speed Encoder input is not available for VC20XX, VC4018, VC4016 and VCSBC cameras. There is however the possibility of using a software trigger counter with the Trigger In input (see the 2 demo programs “starttrig” and “trigtask”) with a maximum trigger input frequency of 10kHz (depending on interrupt latency caused by executed programs).

1.2 Connecting a 3 channel encoder to the camera trigger interface

The following table shows the connection of a 3 channel encoder at the trigger interface of a VC4XXX camera.

Pin Allocation Trigger Interface Cable / incremental Encoder:



Pin	Signal	Core Colors trigger cable	Encoder
1	V24 TxD Out	green	0+ (N or Zero pulse)
2	+ 5V Out	brown	UB (5V power supply)
3	GND	white	GND
4	V24 RxD In	pink	B+
5	Trigger Out	grey	NC
6	Trigger In	yellow	A+

2 Programming of the Encoder Interface

Technical data of high speed encoder counter

- Max counter value: 16777215 (24 bit)
- Max signal frequency: 25 MHz
- Detection of rotation direction – included when channel A+ and B+ connected
- Evaluation of 0+ or N signal for synchronization with encoder angle.

Operation Steps for using the Encoder:

The following steps are usually required using the encoder interface:

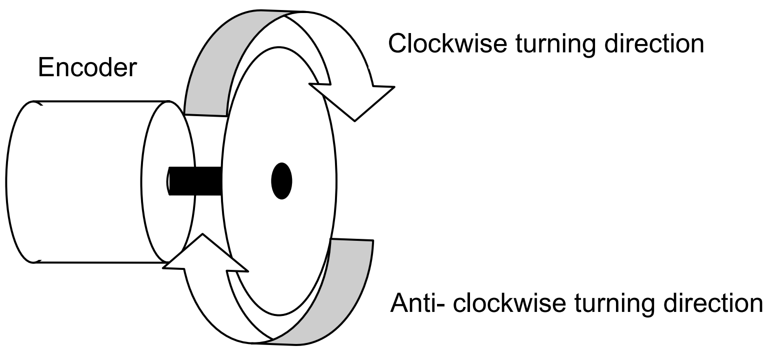
1. Switching the camera to “Encoder Mode”
2. Setting the default turning direction during image acquisition
3. Setting Counter- and Reload target Values
4. Selecting Zero (N) pulse mode
5. Activate counter and start image acquisition
6. Increment Reload Value during operation if necessary
7. Switch back to “Serial Mode” when finished operation

Programming of the steps mentioned above:

1. The following macro enables the encoder interface:

   INTERFACE_MODE(ENCODER); // Encoder mode (RS232 disabled)
2. The following macros set the turning direction for image acquisition:

   The counter increments upwards if the turning direction matches the trigger input signal settings:
   TRIGINP_NEG() means: incrementation during clockwise rotation
   TRIGINP_POS() means: incrementation during anti- clockwise rotation.


Encoder
Clockwise turning direction
Anti- clockwise turning direction

3. Before setting the Counter- and Reload target values it is advisable to disable the counter:

   ENC_DISABLE_CNT(); // disable counter
   The following macros set the Counter- and Reload target values:
   ENC_WRITE_CNT(100); // set Counter value
   ENC_WRITE_RELOAD(250);// set Reload value
   The macros shown activate the first image acquisition at counter value 100 and the second one at counter value 250.

4. Calling one of the following macros resets the counter to 0 at either every turn (N pulse) or only at the first Zero pulse occurrence:

   ENC_WAIT_N(); // wait one time for next 0+ pulse
   ENC_WAIT_N_TRIG(); // wait every time for next 0+ pulse

5. Enabling the counter –incoming signals increment the counter after calling this macro:

   ENC_ENABLE_CNT(); // enable inputs

6. Incrementing the reload value for setting the next counter target value if more than 2 images have to be taken after the counter reset with ENC_WAIT_N() or ENC_WAIT_N_TRIG().

   The following Image acquisition loop increments the target Reload value by 256, taking 4 images during one encoder turn with 1024 pulses:

   ```
   while(condition);
   {
   tenable(); //or capture request for hardware controlled image acquisition – see trigin.c
   ENC_WRITE_RELOAD(Reload);
          Reload = Reload +256;
          if(Reload>790) Reload =0;
   …
   }
   ```

   If incrementing the reload value indefinitely adjust to the counter overrun at 16777215 (24 bit) when incrementing or 0 when turning the encoder in negative direction!

7. Switching the camera back into serial mode – only required if the RS232 is used.

   INTERFACE_MODE(SERIAL);// Serial mode (RS232 enabled)

2.1 Summary of the Encoder Macros

Macro / Function	**Explanation**
INTERFACE_MODE(ENCODER)	Setting the camera to “encoder mode”, this disables the serial interface (RS232). When this macro has not been called, only the Trig In (A+) signal is evaluated (no directional information, no 0+ signal).
INTERFACE_MODE(SERIAL)	Disable “encoder mode” and switching the camera back to “serial mode”.
TRIGINP_POS()	Trigger counter increments with anti- clockwise encoder rotation. Turning the encoder clockwise decrements the counter. Images are only taken while the counter is incrementing (locking mechanism)!
TRIGINP_NEG()	Trigger counter increments with clockwise encoder rotation.
ENC_DISABLE_CNT()	Disable encoder counter: Further encoder signals are ignored until the counter has been enabled again. This function makes it possible to initialize the encoder counter undisturbed form incoming signals.
ENC_ENABLE_CNT()	Enable encoder counter: From execution of this function, the encoder counter starts incrementing / decrementing.
ENC_WRITE_CNT(X)	Setting the current counter target value for the first image acquisition. The counter target value defaults to 1 if not set.
ENC_WRITE_RELOAD(X)	Setting the reload target value for the second image acquisition and all further image acquisitions if not altered during acquisition. After the first image acquisition, this reload value is written into the counter target register. Unless the reload value is changed, the counter target value is set with this reload value after each image acquisition. The reload value defaults to 1 if not set. This means image acquisition with every encoder counter signal (standard trigger mode).
ENC_WAIT_N()	If ENC_WAIT_N() has been called, the counter value is reset to 0 only the next time the “0+” signal has been received. This function references the encoder angle only at the first rotation.
ENC_WAIT_N_TRIG()	ENC_WAIT_N_TRIG() references the encoder angle with every rotation.
ENC_READ_CNT()	Read out current counter value of encoder counter – since this function is not hardware controlled it might not return an “up to date” counter value depending on the signal frequency!

2.2 Application 1 Vision 2005 trade show demo

- Hardware with only one encoder channel input (or hardware with 3 inputs but use of serial interface).

1. Using only one encoder channel input (A+) on Trigger In (Pin 6) ,5V high
2. Power supply Encoder with 24V
3. 0+ (N) signal input on PLC input 0, 21.5 V high (Ub-2.5V), resetting the counter at slow rotation (machine start) since PLC input slow in comparison.

This configuration is sufficient *as long as the machine rotation is only in one direction and does not stop* – for instance a constantly running conveyor belt. The use of all 3 encoder signals is recommended for safe operation (Application 2 and 3). Synchronization of the encoder angle is required at every machine start, since with evaluating only one encoder channel, the counter may change even at standstill (bouncing signal).

Code sample for encoder synchronized image acquisition:

…	
setPLC0();	Start motor of rotating discs
while((inPLC()&0x01)==0);	Waiting for first 0+ (resp. N) signal during slow machine start-up, synchronizing the rotating discs with the image acquisition.

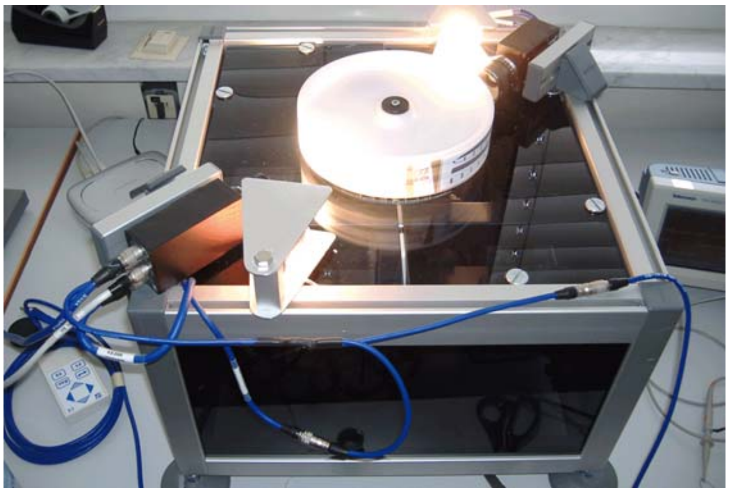
ENC_DISABLE_CNT();
ENC_WRITE_CNT(256);
ENC_WRITE_RELOAD(256);
	Setting the counter and reload value with counter disabled.

ENC_ENABLE_CNT();	Enable the counter – start decrementing the counter.

while(Run)
{	

tenable();
while(!trdy())
…
	Usually the tenable takes an image with every trigger signal input. Here: hardware controlled image acquisition every time the counter value matches the next counter target value

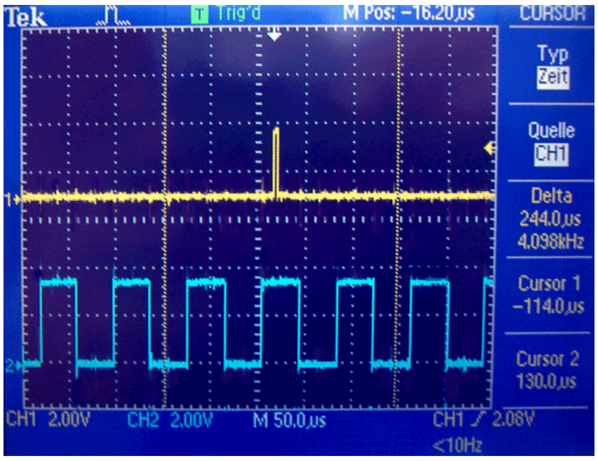
2.3 Application 2: New Demo Turntable



- Application 2 uses the new demo turntable using all 3 encoder channels. The TTL encoder is mounted directly to the motor shaft, the encoder power supply is done via the 5V out of the trigger interface (encoder directly connected via trigger cable).
- In this application the counter resets to 0 at every Zero pulse (N resp. 0+).
- Since 4 images have to be taken at the disc circumference, the “Reload value” is incremented by 256 (1024 pulse/ rev. encoder) after each image acquisition, setting the target value for the image after the next.
- Since the initial counter value is set to 0 and the Reload value to 256, the first image acquisition takes place at counter = 0 and the second at counter = 256, since the images are aligned with the Zero pulse angle. Adjusting to any offset can simply be done by changing these initial values.
- Incrementing the Reload value inside the image loop (bold section in code below) adjusts the position of the next image acquisition.

Note:

- Although the encoder counter and image acquisition is hardware controlled, incrementing the Reload value is a time critical process! Care must be taken that the counter does not “overtake” the Reload value! In this case, no further image acquisition will take place until the next turn (counter reset). Ensure the image acquisition program runs at highest priority if running several parallel tasks!
- Tests at 45 frames per second, >10kHz encoder signal with the above program have shown that the image acquisition as worked reliably.



Oscilloscope screenshot with Encoder A channel and trigger output showing the time of image exposure at 5 μs shutter, trailing the previous pulse with about 50 μs constant capture delay (VC4038).

2.3.1 Application 2 demo program with counter reset at every turn (Zero pulse):

This program is attached to an “**Knowledge Base**” article about the “New Highspeed Encoder Interface” – please search the “**Knowledge Base**” for the keyword “**Encoder**”.

Program description:

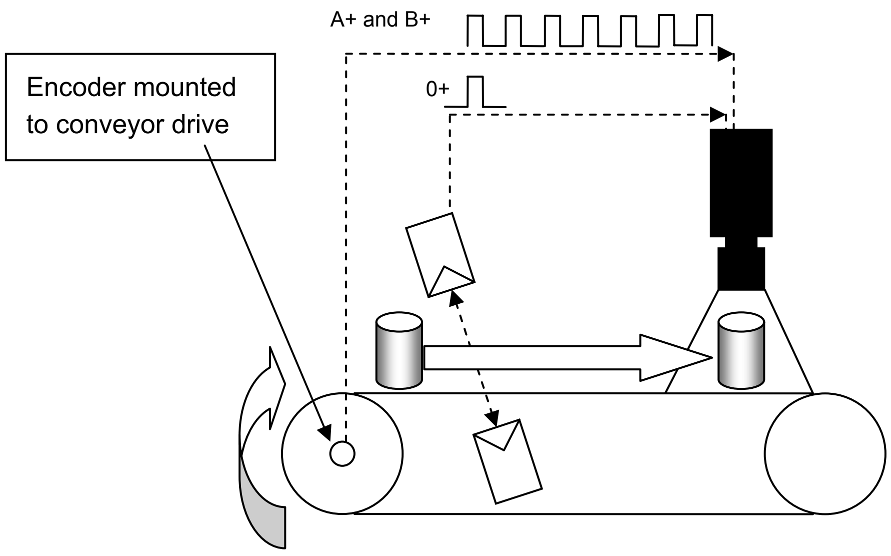
This program demonstrates the use of the encoder interface. 4 images are taken during one turn at counter 0, 256, 512 and 768. The printout shows the actual counter value, the currently set reload value and the current MSEC value. In this program the counter resets every revolution at the "N" (Zero or 0+) pulse, calling the macro "ENC_WAIT_N_TRIG();" at the beginning of the program.

```
/*******************************************************************/
#include <vcrt.h>
#include <vclib.h>
#include <macros.h>
#include <sysvar.h>
#include <flib.h>
#include <licence.h>            // internal VC header including init_licence() functions, see "Getting Started VC", section 4
/*******************************************************************/
void main(void)
{
I32 Run = 1, TrackNr=0;
U32 Reload = 256;
char c;

INIT_LICENCE();                 // internal VC macro initialising licences
shutter(10);                    // setting the shutter to 10 microseconds

TRIGINP_NEG() ;                 // clockwise rotation
TRIGOUT_EXP();                  // activate trigger output during exposure for ozilloscope screenshot
TRIGOUT_POS();                  // setting trigger output to active high

tpict();                        // take image to end live mode

INTERFACE_MODE(ENCODER);        // Encoder mode (RS232 disabled)
ENC_DISABLE_CNT();              // disable counter
//ENC_WAIT_N();                 // wait one time for next 0+ pulse
ENC_WAIT_N_TRIG();              // wait every time for next 0+ pulse
ENC_ENABLE_CNT();               // enable inputs
ENC_WRITE_CNT(0);               // set Counter value
ENC_WRITE_RELOAD(Reload);       // set Reload value

setPLC0();                      // this starts the motor - clockwise rotation, when looking towards the motor


while(Run)                      // Image acquisition loop – see “trigin.c” in “Programming Tutorial Basics”
        {
        TrackNr= capture_request(getvar(EXPCNT), getvar(GAIN),(int*)ScrGetCaptPage,1); // calling image acquisition
        while(TrackNr != getvar(IMGREADY));  // wait for image completion (only works if cycle time > exp. + transfer time)
        {
                ENC_WRITE_RELOAD(Reload); // Writing the Reload value for the image after the next
                Reload = Reload +256;   // Incrementing the Reload value
                if(Reload>790) Reload =0;

                if(kbhit())             // check for keyboard input
                {
                c=rs232rcv();
                if(c==0x1B)             // check for "ESC"
                        {
                        while(cancel_capture_rq());
                        Run = 0;        // deleting last capture request
                        }
                }

                printf("%08d, %d, %d\n",ENC_READ_CNT(), Reload, getvar(MSEC));
                time_delay(1);          // waiting time to slow printing
        }
         }
 INTERFACE_MODE(SERIAL);        // Serial mode (RS232 enabled)
 vmode(0);                      // switching camera back to live mode
 resPLC0();                     // turn off the motor
 }
 /*******************************************************************/
```

3 Application 3: Synchronization with Conveyor belt

In order to synchronies the image acquisition with a conveyor or linear axis only minor changes are required to the program and setup as shown in Application 2:

- Connecting a TTL level light barrier to Zero pulse input Pin 1
- Omitting to increment the Reload value in above program, since the target counter value is constant.
- This setup ensures a reliable image acquisition independent of the conveyor speed.



Application example: Synchronizing the image acquisition on a conveyor belt

- The implementation of two further hardware counter is under preparation. This will allow hardware interrupt controlled sorting of good/ bad parts downstream of the camera station.

It's no trick…
it's a vision system

Visit the Vision Components site **www.vision-components.com** for further information and documentation and software downloads:

Web Site Menu Links	**Content**
Contact	Distributor list / Enquiry forms
Home	Latest News from VC
Our Company	VC Company Information
News and Events	Trade Show dates VC Publications Sign in for free VC Seminars
VC Network	Description of Partner Companies Application Overview 3rd Party Hard- and SW Products for VC Smart Cameras
Products VC Smart Camera Overview	Product Overview: VC44XX High End Camera Series VC40XX Standard Camera Series VC4016 / 18 Entry Level Cameras VC4002L Line Scan Camera VCSBC Single Board Cameras VC20XX Smart Cameras VCSBC Board Cameras VCM + Viscube Camera Sensors
VC Smart Camera Software VC Software Development Kit Ti:	VCRT Operating System VCLIB Image Processing Library
VC Special Libraries:	M200 Data Matrix Code Reader VCOCR Text Recognition Library Color Lib
Support: Support News (User Registration required)	Tech News – new SW and Documentation
Knowledge Base / FAQ (User Registration required)	Searchable FAQ Database with programming Examples and Demo Code
Download Area Public Download Area (free Access) Registered User Area (User Registration required) Customer Download Area (User- and SW License Registration required) RMA Number Form	Download of: - Product Brochures 🇬🇧 🇩🇪 🇫🇷 - Camera Manuals - Getting Started 🇬🇧 🇩🇪 🇫🇷 - Programming Manuals - Training Manuals and Demo Code - Software Updates - Demo Code Form for Allocation of Repair Numbers.