AccelWare IP cores provide a direct path to hardware implementation for complex MATLAB® toolbox and built-in functions. AccelWare cores deliver synthesizable, pre-verified DSP functions that enable true, top-down MATLAB architectural synthesis of FPGAs and ASICs. AccelWare IP includes Building Block, Advanced Math, Signal Processing and Communications toolkits.

# Matrix Factorization (QR method)

Factorizes a real-valued, possibly rectangular, input matrix into an upper triangular R matrix and an orthogonal matrix Q such that Q*R = A. (For applications where matrix inversion is required, see the data sheet for the **Matrix Inverse – QR method**).

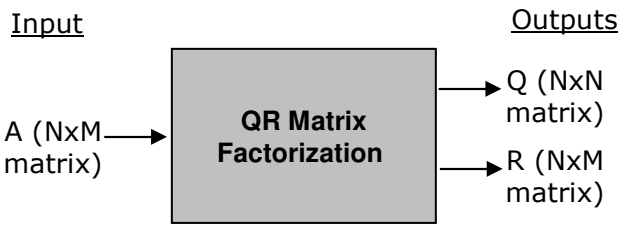


Figure 1: Matrix Factorization Block Diagram

The Matrix Factorization model uses an algorithm based on Givens Rotations (GR) to produce the triangular (R) and orthogonal (Q) factors. The Givens Rotations are implemented in their 'conventional' form using a COordinate Rotation DIgital Computer (CORDIC) for the vector rotations required to null elements below the diagonal and produce the upper triangular matrix R [1].

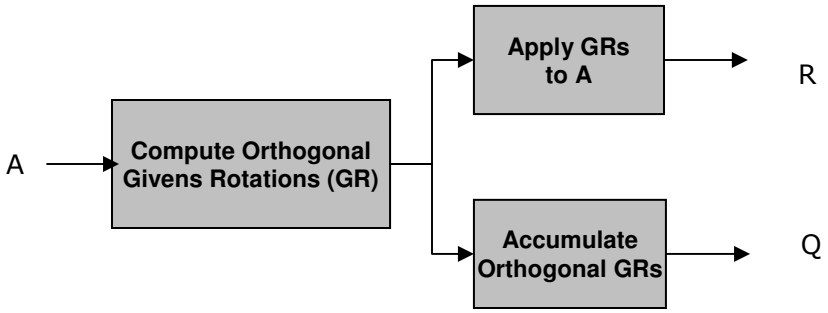


Figure 2: QR Factorization Block Diagram

1] John G. Proakis et al. "Advanced Digital Signal Processing," Macmillan Publishing Company, New York, New York, 1992.

# Matrix Factorization (QR method)

| Input | | | |
|---|---|---|---|
| **Signal Name** | **Signal Description** | **Type** | **Range** |
| A | Input matrix to be factored | Real | 4 to 24 bits (fixed-point representation) |

| Output | | | |
|---|---|---|---|
| **Signal Name** | **Signal Description** | **Type** | **Range** |
| Q | Orthogonal factor matrix | Real | Up to 32 bits (fixed-point representation) |
| R | Upper triangular factor matrix | Real | Up to 32 bits (fixed-point representation) |

| Implementation Parameters | | |
|---|---|---|
| **Parameter Name** | **Parameter Description** | **Range** |
| A quantizer | Input matrix quantization | Two to 24 bits of word-length with possible fractional part (e.g. [16 15]). |
| Implementation Algorithm | Algorithm used for factorization implementation | Conventional Givens Rotations |
| Input Data Type | Input matrix data type | Real |
| Matrix Rows | Input matrix number of rows | Integer in range 4 to 64 |
| Matrix Columns | Input matrix number of columns | Integer in range 4 to 64 |
| Input/Output Type | Input/output matrix dimension representation | 1-D or 2-D |
| Rotation Angle Precision | Number of bits for Givens rotations angle computation | Integer in range 4 to 24; auto sets value to input matrix word length |
| Q Matrix Precision | Bits for Q matrix quantization | Integer in range 4 to 24; auto sets value to input matrix word length |
| R Matrix Precision | Bits for R matrix quantization | Integer in range 4 to 24; auto sets value based on input matrix word length |
| Resource Sharing | Resource-shared implementation option | Yes |

### Input Matrix Quantization
The number representation of the input is defined by the input matrix quantization. The *qr_factor* accepts real-valued, fixed-point input data with quantization properties defined by this parameter.

### Implementation Algorithm
The implementation algorithm used for matrix factorization is based on conventional Givens Rotations with CORDIC.

### Input Data Type
The input matrix must be real-valued.

### Matrix Size
The matrix size is defined by the Matrix Rows and Matrix Columns parameters. The input matrix can be square or rectangular depending on the values specified for these parameters.

### Input/Output Data Type
The *qr_factor* model can be generated to accept input and generate output matrices as 1-D or 2-D arrays.

### Rotation Angle Precision
This parameter defines the precision, in number of bits, used in the calculation of the rotation angle in Givens Rotations.

### Q Matrix Precision
This parameter defines the precision, in number of bits, for the number representation of the output Q matrix. Selecting *auto* sets the precision to the value of the input matrix word-length.

### R Matrix Precision
This parameter defines the precision, in number of bits, for the number representation of the output R matrix. Selecting *auto* sets the precision based on the input matrix word-length.

## Hardware Interfacing
A synthesizable AccelWare MATLAB model will typically be a design module that is part of a larger design on a chip. The flow of data into and out of the hardware ports is controlled by a protocol called DAP (Data Accept Protocol). Synthesizing the *qr_factor* model with AccelChip in a stand-alone fashion will produce a Matrix Factorization hardware block with DAP interface signals ready for integration into a larger system. The following gives a description of DAP interface protocol.

### Global Signals
The hardware module has one Clock input and one global Reset. Data transfers on each port are synchronized to the Clock. The global Reset returns all registers and flip-flops to a known state.

### Input Synchronization Signals
**ND (NewData)** -This signal is controlled by the external design and indicates that data on the input data bus is valid. This causes the receiving device (the hardware module) to capture the data on the rising edge of the next clock cycle.

**RFND (ReadyForNewData)** -This signal is controlled by the hardware module and indicates that the module is ready to capture new data from the input bus. When the module sets RFND low, the external design should immediately stop sending new data. If the hardware module holds RFND constantly high, then new data will be captured on every clock cycle provided the sending device can send data that fast.

### Output Synchronization Signals
**Done** -This signal is controlled by the hardware module and indicates that data on the output data bus is valid. Once Done is set high, it will remain high until the receiving device acknowledges the data capture by setting DA high. If the external design holds DA constantly high, then the hardware module will send data at the maximum possible rate, as governed by the module clock frequency and the latency of the computing algorithm.

**DA (Data Accept)** - This signal is controlled by the external design and indicates that the data on the output bus has been captured. If the external design holds DA constantly high, then the hardware module will send data out at the maximum rate possible, as governed by the module clock frequency and the latency of the computing algorithm.

| Signal | Direction | Description |
|---|---|---|
| Clock | Input | Clock input |
| Reset | Input | Reset input |
| A_flat | Input | Input matrix data |
| RFND_A | Output | Ready for new data |
| ND_A | Input | New input data valid |
| Q_write_flat | Output | Q Output matrix data |
| Done_Q_write | Output | Q Done indication |
| DA_Q_write | Input | Q Data accepted indication |
| R_write_flat | Output | R Output matrix data |
| Done_R_write | Output | R Done indication |
| DA_R_write | Input | R Data accepted indication |

Figure 3: DAP Signals in *qr_factor*

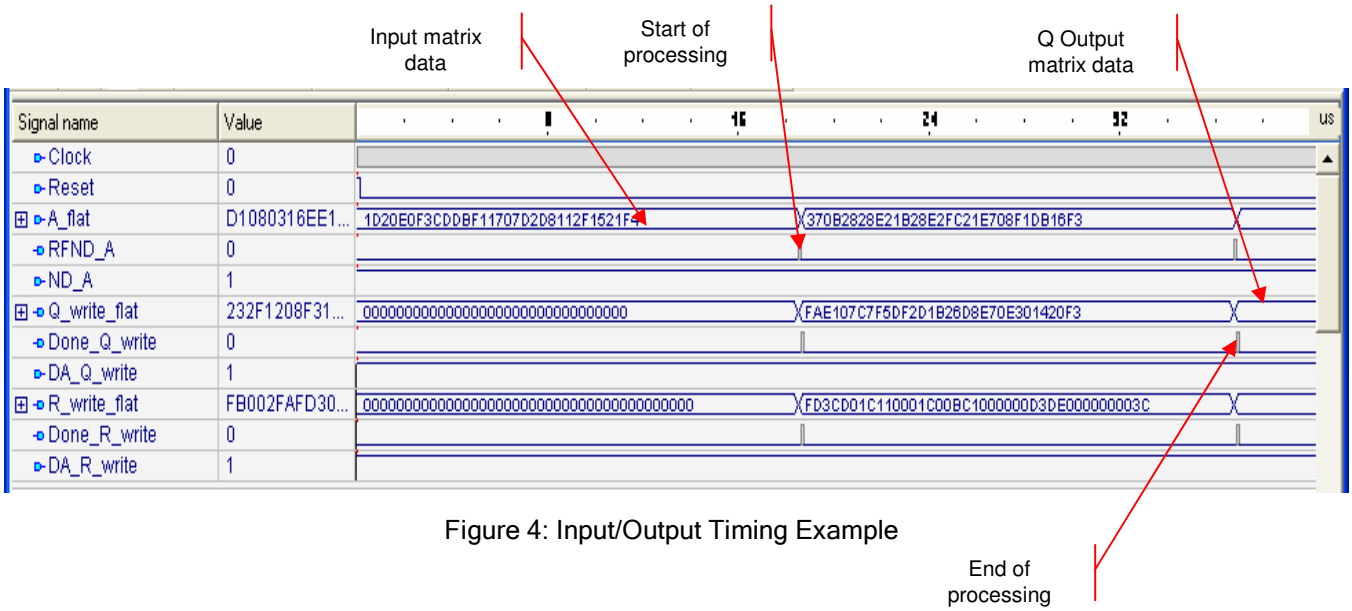


Figure 4: Input/Output Timing Example

## Differences in Operation between AccelWare *qr_factor()* and MATLAB *qr()*

The MATLAB *qr* function can operate on matrices with complex data producing an upper triangular R and a unitary Q. The AccelWare *qr_factor* currently operates on real-valued matrices only.

The MATLAB *qr* function can operation on full and sparse matrices. The AccelWare *qr_factor* operates on full matrices only.

## Ordering Information

The AccelWare *qr_factor* core is included in the AccelWare Advanced Math Toolkit (AccelChip part number **AWAMT**) and is provided as an option to the AccelChip DSP Synthesis product (AccelChip part number **ACDSP**).

For further information on availability, contact your local AccelChip sales representative or send email to sales@accelchip.com.

**AccelChip Incorporated**
1900 McCarthy Blvd., Suite. 204, Milpitas, CA 95035 phone (408) 943 0700 option 1 fax (408) 943 0661