Clarke & Park Transforms on the TMS320C2xx

Application Report
Literature Number: BPRA048

TEXAS INSTRUMENTS

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain application using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

Table of Contents

1. Overview

Clarke and Park transforms are used in high performance drive architectures (vector control) related to permanent magnet synchronous and asynchronous machines. In this paper, the user will find functions to easily implement Clarke and Park transforms to his application.

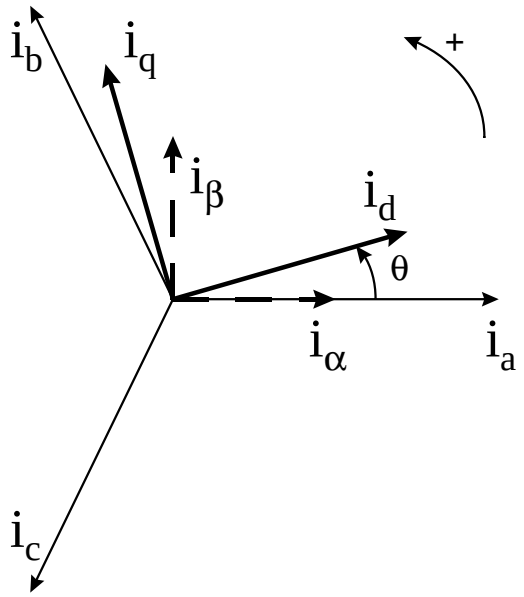
Through the use of the Clarke transform, the real (Ids) and imaginary (Iqs) currents can be identified. The Park transform can be used to realize the transformation of the Ids and the Iqs currents from the stationary to the moving reference frame and control the spatial relationship between the stator vector current and rotor flux vector.

2. Clarke and Park transforms in the Field Orientated Control (FOC)

The FOC consists of controlling the components of the motor stator currents, represented by a vector, in a rotating reference frame *d,q* aligned with the rotor flux. The vector control system requires the dynamic model equations of the induction motor and returns the instantaneous currents and voltages in order to calculate and control the variables.

The electric torque of an AC induction motor can be described by the interaction between the rotor currents and the flux wave resulting from the stator currents induction. Since the rotor currents cannot be measured with cage motors, this current is replaced by an equivalent quantity described in a rotating system coordinates called *d,q* following the rotor flux.

The Clarke transform uses three-phase currents i_a, i_b and i_c to calculate currents in the two-phase orthogonal stator axis: i_α and i_β. These two currents in the fixed coordinate stator phase are transformed to the i_{sd} and i_{sq} currents components in the d,q frame with the Park transform. These currents i_{sd}, i_{sq} and the instantaneous flux angle ρ , calculated by the motor flux model, are used to calculate the electric torque of an AC induction motor.



Stator current in the d,q rotating reference frame
and its relationship with the a,b and c stationary reference frame.

After such a transformation, the stator variables (currents and angle) are translated into a flux model. This flux model is compared with the reference values and updated by a PI controllers. After a back transformation from field to stator coordinates, the output voltage will be impressed to the machine with Pulse Width Modulation (PWM).

3. Mathematical consideration.

3.1 Mathematical Clarke transform.

The mathematical transformation called Clarke transform modifies a three-phase system to a two-phase orthogonal system:

$$i_\alpha = \frac{2}{3} \cdot i_a - \frac{1}{3}(i_b - i_c)$$

$$i_\beta = \frac{2}{\sqrt{3}}(i_b - i_c)$$

$$i_o = \frac{2}{3}(i_a + i_b + i_c)$$

with i_α and i_β components in an orthogonal reference frame and
i_o the homopolar component of the system.

In many applications, the homopolar component is absent or is less important. In this way, in absence of homopolar component the space vector $u = u_\alpha + ju_\beta$ represents the original three-phase input signal.

Consider now a particular case with i_α superposed with i_a and $i_a + i_b + i_c$ is zero, in this condition i_a, i_b and i_c can be transformed to i_α and i_β with following mathematical transformation:

$$i_\alpha = i_a$$

$$i_\beta = \frac{1}{\sqrt{3}} \cdot i_a + \frac{2}{\sqrt{3}} i_b$$

$$i_a + i_b + i_c = 0$$

3.2 Mathematical Park transform.

The two phases α, β frame representation calculated with the Clarke transform is then fed to a vector rotation block where it is rotated over an angle θ to follow the frame d,q attached to the rotor flux.

The rotation over an angle θ is done according to the formulas:

$$i_{sd} = i_\alpha \cdot \cos(\theta) + i_\beta \cdot \sin(\theta)$$
$$i_{sq} = -i_\alpha \cdot \sin(\theta) + i_\beta \cdot \cos(\theta)$$

3.3 Mathematical Inverse Park and Clarke transforms.

The vector in the d, q frame is transformed from d, q frame to the two phases α, β frame representation calculated with a rotation over an angle θ according to the formulas:

$$i_\alpha = i_{sd} \cdot \cos(\theta) - i_{sq} \cdot \sin(\theta)$$
$$i_\beta = i_{sd} \cdot \sin(\theta) + i_q \cdot \cos(\theta)$$

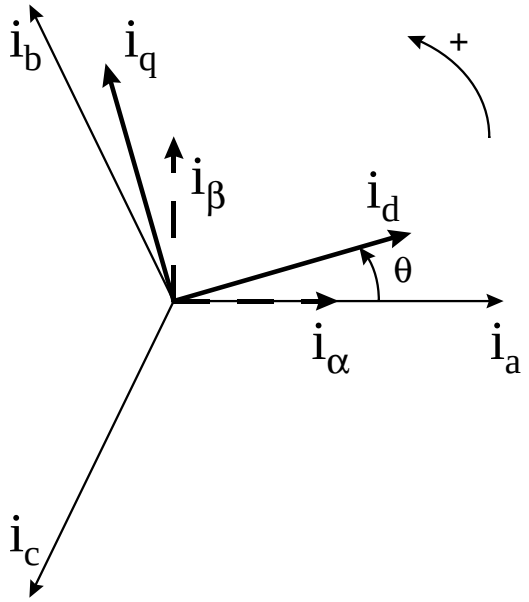
The modification from a two-phase orthogonal α, β frame to a three-phase system is done by the following equations:

$$i_a = i_\alpha$$
$$i_b = -\frac{1}{2}\cdot i_\alpha + \frac{\sqrt{3}}{2}\cdot i_\beta$$
$$i_c = -\frac{1}{2}\cdot i_\alpha - \frac{\sqrt{3}}{2}\cdot i_\beta$$

3.4 Transforms summary.

Park	***Inverse Park***
a, b, c -> α, β	**d, q -> α, β**
$i_\alpha = \frac{2}{3}\cdot i_a - \frac{1}{3}(i_b - i_c)$ $i_\beta = \frac{2}{\sqrt{3}}(i_b - i_c)$ $i_o = \frac{2}{3}(i_a + i_b + i_c)$ $\Rightarrow$ $i_\alpha = i_a$ $i_\beta = \frac{1}{\sqrt{3}}\cdot i_a + \frac{2}{\sqrt{3}} i_b$ $i_a + i_b + i_c = 0$	$i_\alpha = i_{sd}\cdot\cos(\theta) - i_{sq}\cdot\sin(\theta)$ $i_\beta = i_{sd}\cdot\sin(\theta) + i_q\cdot\cos(\theta)$
α, β -> d, q	**α, β -> a, b, c**
$i_{sd} = i_\alpha\cdot\cos(\theta) + i_\beta\cdot\sin(\theta)$ $i_{sq} = -i_\alpha\cdot\sin(\theta) + i_\beta\cdot\cos(\theta)$	$i_a = i_\alpha$ $i_b = -\frac{1}{2}\cdot i_\alpha + \frac{\sqrt{3}}{2}\cdot i_\beta$ $ic = -\frac{1}{2}\cdot i_\alpha - \frac{\sqrt{3}}{2}\cdot i_\beta$

With vectors conventions:



4. Clarke and Park implementation on the C2xx

4.1 Conventions

Two different versions of each function are presented in this document, fully C compatible functions and assembly calling functions.

Different examples of assembly and C program calling Park and Park_inverse transforms are in the Annexe.

4.1.1 Fully C-compatible functions

Fully C compatible functions use C convention to use the stack for parameters passed to the functions. Parameters returned by functions are passed by pointer. Responsibilities of a C called function are managed. Stack pointer AR1 is well positioned and the return address of the hardware stack is popped in case of a C interrupt events using C-function features (stack). The frame pointer is not modified. Register AR6/AR7 are not used.

4.1.2 Assembly compatible functions

Assembly compatible functions do not use their own variables, variables used are in a stack.

Arguments are passed by the stack and the AR1 point to the stack just after the last argument. In return from function, results are in the stack. Register AR0, AR6 and AR7 are not modified.

4.2 Functions

4.2.1 Park assembly compatible

Function Park with Clarke and Park transforms is in the annexe with a main assembly example. Conventions to interface with this function are:

- Input:
  parameters are in the stack pointed by AR1 (AR1 point the address in the stack just after Angle parameter) with the following order:
  - current ia -32768<ia<32767
  - current ib -32768<ib<32767
  - ANGLE parameters

  The value of this angle is unsigned:

  0 º <-> 0000h
  90 º <-> 4000h
  180 º <-> 8000h
  240 º <-> C000h

- Output:
  i_{sd} , i_{sq}, sin(angle) and cos(angle) are in the stack in this order with the same fixed point format than i_a, i_b in input. AR1 point on cos.

COS and SIN calculation are done by a single function for better optimization with Table Look-up and Linear Interpolation (Cf: application note “Sine & Cosine on the TMS320C2xx”).

Calculations are done with fixed point instruction to optimize the time calculation. The dynamic used in calculation fit with maximum precision and overflow is managed.

4.2.2 Inverse Park assembly compatible

Two Functions Inverse Park assembly compatible are in the annexe. One functions recalculates the sine and the cosine of angle in case these values are not saved in Park function return. The second functions use sine and cosine passed in parameters for calculation.

Calculations are done with fixed point instructions to optimize the time calculation. The dynamic used in calculation fit with maximum precision overflow is managed.

4.2.2.1 Inverse Park assembly compatible with cos, sin calculation

Conventions to interface with the first function are:

- Input:
  parameters are in the stack pointed by AR1 (AR1 points to the address in the stack just after Angle parameter) with the following order:
  - current isd -32768<isd<32767
  - current isq -32768<isq<32767
  - ANGLE parameter

  The value of this angle is unsigned:

  0 º <-> 0000h
  90 º <-> 4000h
  180 º <-> 8000h
  240 º <-> C000h
- Output: i_a, i_b, i_c are in the stack in this order with the same fixed point format than i_d, i_q in input. AR1 points on the address in the stack just after ic.

COS and SIN calculation are done by a single function for better optimization with Table Look-up and Linear Interpolation (Cf: application note “Sine & Cosine on the TMS320C2xx”).

4.2.2.2 Inverse Park without cos, sin calculation

Conventions to interface with the second function are:

- Input:
  parameters are in the stack pointed by AR1 (AR1 points to the address in the stack just after COS parameter) with the following order:
  - current isd -32768<isd<32767
  - current isq -32768<isq<32767
  - SIN(angle) in the same format than in Park return Q15.
  - COS(angle) in the same format than in Park return Q15.

  The value of this angle is unsigned:

  0 º <-> 0000h
  90 º <-> 4000h
  180 º <-> 8000h
  240 º <-> C000h
- Output: i_a, i_b, i_c are in the stack in this order with the same fixed point format than i_d, i_q in input. AR1 point on the address in the stack just after ic.

4.2.3 Park C compatible

Function Park with Clarke and Park transforms is in the annexe with a main C example. Conventions to interface with this function are passed in integer. Output of the function are passed back by pointer.

VOID PARK (int angle, int ib, int ia, int *iq, int *id, int *cos, int *sin)

A second function which does not return by pointer sine and cosine parameters is in the annexe. The declaration of this function is:

VOID PARK (int angle, int ib, int ia, int *iq, int *id)

COS and SIN calculations are done by a single function for better optimization with Table Look-up and Linear Interpolation (Cf: application note “Sine & Cosine on the TMS320C2xx”).

Calculations are done with fixed point instruction to optimize the time calculation. The dynamic used in calculation fit with maximum precision, overflow is managed.

4.2.4 Inverse Park C compatible

Function Inverse Park is in the annexe with a main C example. Conventions to interface with this function are Input parameters are passed in integer. Output of the function are passed back by pointer.

VOID INV_PARK (int cos, int sin, int iq, int id, int *ia, int *ib, int *ic)

A second function which calculates sine and cosine values is in the annexe. The declaration of this function is:

VOID INV_PARK (int angle, int iq, int id, int *ia, int *ib, int *ic)

In case of the second function, COS and SIN calculation are done by a single function for better optimization with Table Look-up and Linear Interpolation (Cf: application note “Sine & Cosine on the TMS320C2xx”).

Calculations are done with fixed point instruction to optimize the time calculation. The dynamic used in calculation fit with maximum precision, overflow is managed.

4.3 Processor utilization (maximum)

4.3.1 Park

Function	Cycles	Execution Time
Clarke_Park + COS_SIN in line (assembly main)	44+53=97	4.85µs
Clarke_Park (COS_SIN included) Fully C compatible	64+53=117	5.85µs
Clarke_Park (COS_SIN included) return sine and cosine Fully C compatible	72+53=125	6.25µs

4.3.2 Inverse Park

Function	Cycles	Execution Time
Inverse Park + COS_SIN in line (assembly main)	48+53=101	5.05µs
Inverse Park without cos/sin calculation in line (assembly main)	48	2.4µs
Inverse Park Sine and cosine in parameters Fully C compatible	74	3.70µs
Inverse Park + COS_SIN in line Fully C compatible	73+53=126	6.30µs

4.3.3 Park + Inverse Park

Function	Cycles	Execution Time
Park + Inverse Park with cos/sin saving in line (assembly main)	97+48=145	7.25µs
Park + Inverse Park Fully C compatible	117+74=191	9.55µs

4.4 Memory utilization

4.4.1 Park

Function	ROM (words)	Stack levels	Registers used	RAM (words)
Clarke_Park + COS_SIN in line (assembly main)	42+60+125=227	7	1	in stack
Clarke_Park + COS_SIN without sine and cosine in parameters fully C compatible	58+60+125=243	10	3	in stack
Clarke_Park + COS_SIN with sine and cosine in parameters fully C compatible	64+60+125=249	12	3	in stack

4.4.2 Inverse Park

Function	ROM (words)	Stack levels	Registers used	RAM (words)
Inverse Park with cos/sin in parameters (assembly main)	46	5	1	in stack
Inverse Park with cos/sin in parameters fully C compatible	66	9	4	in stack
Inverse Park with cos/sin calculation (function COS_SIN shared with Park) fully C compatible	65	11	1	in stack

4.4.3 Park + Inverse Park

Function	**ROM (words)**	**Stack levels**	**Registers used**	**RAM (words)**
Park + Inverse Park (assembly main)	227+46=273	7	1	in stack
Park + Inverse Park fully C compatible	243+66=309	10	4	in stack

5. Annexe

5.1 Main assembly example to call Park and inverse Park function without cos/sin calculation in inverse Park

```
;*****************************************************************
;*File Name:     M_Park.asm                                       *
;*Project:       DMC Mathematical Library                         *
;*Originator:    Pascal DORSTER (Texas Intruments)                *
;*                                                                *
;*Description:   Very simple main which call Park                 *
;*               and inverse Park function without cos/sin        *
;*               calculation in inverse Park                      *
;*                                                                *
;*Processor:     C2xx                                             *
;*                                                                *
;*Status:                                                         *
;*                                                                *
;*Last Update:   19 Oct 96                                        *
;*________________________________________________________________*
;*Date of Mod    |          DESCRIPTION                           *
;*---------------|------------------------------------------------*
;*               |                                                *
;*               |                                                *
;*****************************************************************

  .mmregs

  .sect  "vectors"
  b      _c_int0
  b      $

  .global _INV_PARK
  .global _PARK
;*****************************************************************
;* Main routine                                                  *
;*****************************************************************
  .text

_c_int0:
  LAR    AR1,#60h          ;stack in the scratch window
  MAR    *,AR1

  LAC    #0500h            ;ia
  SACL   *+
  LAC    #0400h            ;ib
  SACL   *+
  LAC    #1000h            ;angle
  SACL   *+
                           ;*STACK : ia/ib/angle/X
  CALL   PARK
```

```
                            ;*STACK : isd/isq/sin/COS

.
.                           ;up-date isd & isq
.

mar     *+
                            ;*STACK : isd/isq/sin/cos/X
CALL    INV_PARK
                            ;*STACK : ia/ib/ic/x/X
SBRK    4
LAC     *+                  ;ia
LAC     *+                  ;ib
LAC     *                   ;ic
NOP

.end
```

5.2 Main assembly example to call Park and inverse Park function with cos/sin calculation in inverse Park

```
*************************************************************
*File Name:     M_Park.asm                                  *
*Project:       DMC Mathematical Library                    *
*Originator:    Pascal DORSTER (Texas Intruments)           *
*                                                           *
*Description:   Very simple main which call Park            *
*               and inverse Park function with cos/sin      *
*               calculation in inverse Park                 *
*                                                           *
*Processor:     C2xx                                        *
*                                                           *
*Status:                                                    *
*                                                           *
*Last Update:   19 Oct 96                                   *
*___________________________________________________________*
*Date of Mod    |          DESCRIPTION                      *
*---------------|-------------------------------------------*
*               |                                           *
*               |                                           *
*************************************************************
  .mmregs

  .sect  "vectors"
  b      _c_int0
  b      $

  .global _INV_PARK
  .global _PARK
*************************************************************
* Main routine
*************************************************************
  .text

_c_int0:
  LAR    AR1,#60h
  MAR    *,AR1

  LAC    #0500h           ;ia
  SACL   *+
  LAC    #0f400h          ;ib
  SACL   *+
  LAC    #2000h           ;angle
  SACL   *+
                          ;*STACK : ia/ib/angle/X
  CALL   PARK
                          ;*STACK : isd/isq/sin/COS/isd

  .
```

```
.                       ;up-date isd & isq
.

MAR     *-
LAC     #2000h
SACL    *+              ;angle
                        ;*STACK : isd/isq/angle/X
CALL    INV_PARK
                        ;*STACK : ia/ib/ic/X
SBRK    3
LAC     *+              ;ia
LAC     *+              ;ib
LAC     *               ;ic
NOP

.end
```

5.3 Clarke_Park function for assembly main

```
*************************************************************
*Routine Name:  PARK                                        *
*Project:       DMC Mathematical Library                    *
*Originator:    Pascal DORSTER (Texas Intruments)           *
*                                                           *
*Description:   Clark + Park calculation                    *
*               with COS & SIN saving for inverse calculation*
*               Assembly calling funtion, variables in s/w  *
*               stack.                                      *
*                                                           *
* Calculation:                                              *
*               Ialpha=Ia                                   *
*                Ibeta =1/SQRT(3)*Ia + 2/SQRT(3)*Ib         *
*                with Ia+Ib+Ic=0                            *
*                                                           *
*                isd =Ialpha*COS(angle) + Ibeta*SIN(angle)  *
*                isq =-Ialpha*SIN(angle) + Ibeta*COS(angle) *
*                                                           *
*                                                           *
*Status:                                                    *
*                                                           *
*Processor:     C2xx                                        *
*                                                           *
*Calling convention:                                        *
*        Assembly calling convention with s/w stack         *
*         Input : Ia, Ib in stack value 16-bits signed      *
*              Angle in stack value 0-360 degrees <=>0h-FFFFh*
*         Output : Isd, Isq, Sin(angle), Cos(angle) in stack *
*         Pointed register AR1                              *
*                                                           *
*Stack commentary:                                          *
*        Position of current register in Caps               *
*         Stack at beginning                                *
*               ia/ib/angle/X                               *
*         Stack at return                                   *
*               isd/isq/sin/COS                             *
*                                                           *
*Function called:                                          *
*        COS_SIN function, cf "SINE, COSINE on the C2xx"    *
*         application note                                  *
*                                                           *
*                                                           *
*Last Update:   16 Oct 96                                   *
*___________________________________________________________*
*Date of Mod    |          DESCRIPTION                      *
*---------------|-------------------------------------------*
*               |                                           *
*                                                           *
*************************************************************
  .global       COS_SIN
```

```
  .global       PARK

ONE_BY_SQRT3    .set    1182
TWO_BY_SQRT3    .set    2364

PARK
  SPM    0h
                       ;*STACK : ia/ib/angle/X
  CALL   COS_SIN       ;calculate SIN & COS of angle
                       ;for vector rotation
                       ;*STACK : ia/ib/sin/COS
  SBRK   3             ;*STACK : IA/ib/sin/cos
                       ;calculation ialpha & ibeta with ia, ib
  LT     *+
                       ;*STACK : ia/IB/sin/cos
  MPYK   ONE_BY_SQRT3
                       ;1/sqrt(3)*ia, one_by_sqrt3 in Q11
  LTP    *
  MPYK   TWO_BY_SQRT3
                       ;2/sqrt(3)*ib, two_by_sqrt3 in Q11
  APAC                 ;1/sqrt(3)*ia+2/sqrt(3)*ib
  ADD    #1,10         ;rounding
  SETC   ovm           ;saturation
  RPTK   3
  NORM   *             ;shift left
  SACH   *             ;
  ADDH   *             ;saturation validation
                       ;shift one left with overflow
  SACH   *             ;save ibeta
                       ;ialpha = ia
                       ;*STACK : ialpha/IBETA/sin/cos
  LT     *+
  MPY    *+            ;ibeta*sin(angle)
                       ;*STACK : ialpha/ibeta/sin/COS
  LTP    *
  SBRK   3
                       ;*STACK : IALPHA/ibeta/sin/cos
  MPY    *             ;ialpha*cos(angle)
  ADRK   4
                       ;*STACK : ialpha/ibeta/sin/cos/X
  APAC                 ;isd=ibeta*sin+ialpha*cos
  ADD    #1,14         ;rounding
  SACH   *
  ADDH   *             ;overflow management
  SACH   *-
                       ;*STACK : ialpha/ibeta/sin/COS/isd
  LT     *
  SBRK   2
                       ;*STACK : ialpha/IBETA/sin/cos/isd
  MPY    *+            ;ibeta*cos
                       ;*STACK : ialpha/ibeta/SIN/cos/isd
  LTP    *
  SBRK   2
```

```
MPY    *+              ;ialpha*sin
                       ;*STACK : ialpha/IBETA/sin/cos/isd
SPAC                   ;isq=ibeta*cos-ialpha*sin
ADD    #1,14
SACH   *
ADDH   *
SACH   *+
                       ;*STACK : ialpha/isq/SIN/cos/isd
ADRK   2
                       ;*STACK : ialpha/isq/sin/cos/ISD
LAC    *
SBRK   4
                       ;*STACK : IALPHA/isq/sin/cos/isd
SACL   *
ADRK   3
                       ;*STACK : isd/isq/sin/COS/isd
RET
```

5.4 Inverse Park function without cos/sin calculation for assembly main

```
;***************************************************************
;* Routine Name:  INV_PARK                                     *
;* Project:       DMC Mathematical Library                     *
;* Originator:    Pascal DORSTER (Texas Intruments)            *
;*                                                             *
;* Description:   Inverse Clark + Park calculation             *
;*                without COS & SIN calculation                *
;*                Assembly calling funtion, variables in s/w   *
;*                stack.                                       *
;*                                                             *
;*  Calculation:                                               *
;*                Ialpha = Id*COS(angle)-Iq*SIN(angle)         *
;*                 Ibeta  = Id*SIN(angle)+Iq*COS(angle)        *
;*                                                             *
;*                Ia = Ialpha                                  *
;*                Ib = -1/2*Ialpha+SQRT(3)/2*Ibeta             *
;*                Ic = -1/2*Ialpha-SQRT(3)/2*Ibeta             *
;*                                                             *
;* Status:                                                     *
;*                                                             *
;* Processor:     C2xx                                         *
;*                                                             *
;* Calling convention:                                         *
;*         Assembly calling convention with s/w stack          *
;*         Input : Isd, Isq in stack value 16-bits signed      *
;*                 SIN(angle), COS(angle) in stack value Q15   *
;*          Output : Ia, Ib, Ic in stack                       *
;*          Pointed register AR1                               *
;*                                                             *
;* Stack commentary:                                           *
;*          Position of current register in Caps               *
;*          Stack at beginning                                 *
;*                  isd/isq/sin/cos/X                          *
;*         Stack at return                                     *
;*                  ia/ib/ic/cos/X                             *
;*                                                             *
;* Last Update:   16 Oct 96                                    *
;*_____________________________________________________________*
;* Date of Mod    |          DESCRIPTION                       *
;*----------------|--------------------------------------------*
;*                |                                            *
;*                |                                            *
;***************************************************************

  .global       INV_PARK

SQRT3_BY_2      .set  0ddbh
```

```
INV_PARK
  SPM    0h
  SETC   ovm
  SETC   sxm
                      ;*STACK : id/iq/sin/cos/X
  MAR    *-
                      ;*STACK : id/iq/sin/COS/x
  LT     *            ;calculation ilpha & ibeta with id, iq
  SBRK   3
                      ;*STACK : ID/iq/sin/cos
  MPY    *+           ;id*cos
                      ;*STACK : id/IQ/sin/cos
  LTP    *+
                      ;*STACK : id/iq/SIN/cos
  MPY    *            ;iq*sin
  SPAC                ;id*cos-iq*sin
  ADRK   2
                      ;*STACK : id/iq/sin/cos/X
  ADD    #1,14
  SACH   *
  ADDH   *
  SACH   *-
                      ;*STACK : id/iq/sin/COS/ialpha
  LT     *
  SBRK   2
                      ;*STACK : id/IQ/sin/cos/ialpha
  MPY    *+           ;iq*cos
                      ;*STACK : id/iq/SIN/cos/ialpha
  LTP    *
  SBRK   2
                      ;*STACK : ID/iq/sin/cos/ialpha
  MPY    *+
                      ;*STACK : id/IQ/sin/cos/ialpha
  APAC                ;idsin+iqcos
  ADD    #1,14
  SACH   *
  ADDH   *+
                      ;*STACK : id/iq/IBETA/cos/ialpha
  SACH   *
                      ;Ia, Ib, Ic calculation
  LT     *-
                      ;*STACK : id/IQ/ibeta/cos/ialpha
  MPYK   SQRT3_BY_2   ;Q12
  PAC                 ;SQRT(3)/2*ibeta
  ADD    #1,11
  RPTK   2
  NORM   *
  SACH   *
  ADDH   *
  SACH   *
  ADRK   3
                       ;*STACK : id/cst*ibeta/ibeta/cos/IALPHA
  LAC    *
```

```
SBRK   4
                    ;*STACK : ID/cst*ibeta/ibeta/cos/ialpha
SACL   *
                    ;*STACK : IA/cst*ibeta/ibeta/cos/ialpha
LAC    *+,15        ;1/2*Ialpha
                    ;*STACK : ia/CST*IBETA/ibeta/cos/ialpha
ADD    *+,16        ;ic=-1/2*Ialpha-sqrt(3)/2*Ibeta
                     ;*STACK : ia/cst*ibeta/IBETA/cos/ialpha
NEG
SACH   *-
                    ;*STACK : ia/CST*IBETA/ic/cos/ialpha
LAC    *-,16
                    ;*STACK : IA/cst*ibeta/ic/cos/ialpha
SUB    *+,15        ;-Ialpha/2
                    ;*STACK : ia/CST*IBETA/ic/cos/ialpha
SACH   *+
                    ;*STACK : ia/ib/IC/cos/ialpha
ADRK   2
                    ;*STACK : ia/ib/ic/cos/IALPHA
RET
```

5.5 Inverse Park function with cos/sin calculation for assembly main

```
*************************************************************
*Routine Name:  INV_PARK                                    *
*Project:       DMC Mathematical Library                    *
*Originator:    Pascal DORSTER (Texas Intruments)           *
*                                                           *
*Description:   Inverse Clark + Park calculation            *
*               with COS & SIN calculation                  *
*               Assembly calling funtion, variables in s/w  *
*               stack.                                      *
*                                                           *
* Calculation:                                              *
*               Ialpha = Id*COS(angle)-Iq*SIN(angle)        *
*                Ibeta  = Id*SIN(angle)+Iq*COS(angle)       *
*                                                           *
*               Ia = Ialpha                                 *
*               Ib = -1/2*Ialpha+SQRT(3)/2*Ibeta            *
*               Ic = -1/2*Ialpha-SQRT(3)/2*Ibeta            *
*                                                           *
*Status:                                                    *
*                                                           *
*Processor:     C2xx                                        *
*                                                           *
*Calling convention:                                        *
*        Assembly calling convention with s/w stack         *
*         Input:                                            *
*             Id, Iq in stack value 16-bits signed          *
*             Angle in stack value 0-360 degrees <=>0h-FFFFh *
*         Output :Ia, Ib, Ic in stack                       *
*         Pointed register AR1                              *
*                                                           *
*Stack commentary:                                          *
*         Position of current register in Caps              *
*         Stack at beginning                                *
*                 id/iq/angle/X                             *
*         Stack at return                                   *
*                 ia/ib/ic/X                                *
*                                                           *
*Function called:                                          *
*        COS_SIN function, cf "SINE, COSINE on the C2xx"    *
*        application note                                   *
*                                                           *
*                                                           *
*Last Update:   16 Oct 96                                   *
*  ________________________________________________________ *
*Date of Mod    |          DESCRIPTION                      *
*---------------|-------------------------------------------*
*               |                                           *
*               |                                           *
```

```
*************************************************************

  .global       INV_PARK

SQRT3_BY_2      .set    0ddbh

INV_PARK
  SPM    0h
  SETC   ovm
  SETC   sxm
                        ;*STACK : id/iq/angle/X
  CALL   COS_SIN        ;calculate SIN & COS of angle
                        ;for vector rotation
                        ;*STACK : id/iq/sin/COS
  LT     *              ;calculation ilpha & ibeta with id, iq
  SBRK   3
                        ;*STACK : ID/iq/sin/cos
  MPY    *+             ;id*cos
                        ;*STACK : id/IQ/sin/cos
  LTP    *+
                        ;*STACK : id/iq/SIN/cos
  MPY    *              ;iq*sin
  SPAC                  ;id*cos-iq*sin
  ADRK   2
                        ;*STACK : id/iq/sin/cos/X
  ADD    #1,14
  SACH   *
  ADDH   *
  SACH   *-
                        ;*STACK : id/iq/sin/COS/ialpha
  LT     *
  SBRK   2
                        ;*STACK : id/IQ/sin/cos/ialpha
  MPY    *+             ;iq*cos
                        ;*STACK : id/iq/SIN/cos/ialpha
  LTP    *
  SBRK   2
                        ;*STACK : ID/iq/sin/cos/ialpha
  MPY    *+
                        ;*STACK : id/IQ/sin/cos/ialpha
  APAC                  ;idsin+iqcos
  ADD    #1,14
  SACH   *
  ADDH   *+
                        ;*STACK : id/iq/IBETA/cos/ialpha
  SACH   *

                        ;Ia, Ib, Ic calculation
  LT     *-
                        ;*STACK : id/IQ/ibeta/cos/ialpha
  MPYK   SQRT3_BY_2     ;Q12
  PAC                   ;SQRT(3)/2*ibeta
  ADD    #1,11
```

```
RPTK   2
NORM   *
SACH   *
ADDH   *
SACH   *
ADRK   3
                      ;*STACK : id/cst*ibeta/ibeta/cos/IALPHA
LAC    *
SBRK   4
                      ;*STACK : ID/cst*ibeta/ibeta/cos/ialpha
SACL   *
                      ;*STACK : IA/cst*ibeta/ibeta/cos/ialpha
LAC    *+,15          ;1/2*Ialpha
                      ;*STACK : ia/CST*IBETA/ibeta/cos/ialpha
ADD    *+,16          ;ic=-1/2*Ialpha-sqrt(3)/2*Ibeta
                      ;*STACK : ia/cst*ibeta/IBETA/cos/ialpha
NEG
SACH   *-
                      ;*STACK : ia/CST*IBETA/ic/cos/ialpha
LAC    *-,16
                      ;*STACK : IA/cst*ibeta/ic/cos/ialpha
SUB    *+,15          ;-Ialpha/2
                      ;*STACK : ia/CST*IBETA/ic/cos/ialpha
SACH   *+
                      ;*STACK : ia/ib/IC/cos/ialpha
MAR    *+
                      ;*STACK : ia/ib/ic/COS/ialpha
RET
```

5.6 Main C example to call Park and inverse Park function without cos/sin calculation in inverse Park

```
************************************************************
*File Name:     M_Park.c                                   *
*Project:       DMC Mathematical Library                   *
*Originator:    Pascal DORSTER (Texas Intruments)          *
*                                                         *
*Description:   Very simple C main which call Park         *
*               and inverse Park function without cos/sin  *
*               calculation in inverse Park                *
*                                                         *
*Processor:     C2xx                                       *
*                                                         *
*Status:                                                   *
*                                                         *
*Last Update:   19 Oct 96                                  *
*_________________________________________________________ *
*Date of Mod    |          DESCRIPTION                     *
*-------------|-------------------------------------------*
*               |                                         *
*               |                                         *
************************************************************/

void PARK(int angle,int ib,int ia,int *iq,int *id,int *cos, int
*sin);
void INV_PARK(int cos,int sin,int iq,int id,int *ia,int *ib,int
*ic);

void main()
  {
  int ia,ib,ic,id,iq,angle;
  int cos,sin;



  angle=1000;
  ia=500;
  ib=400;
  PARK(angle,ib,ia,&id,&iq,&cos,&sin);
  .
  .
  .
  INV_PARK(cos,sin,iq,id,&ia,&ib,&ic);

  }
```

5.7 Main C example to call Park and inverse Park function with cos/sin calculation in inverse Park

```
/*************************************************************
*File Name:     M_Park.c                                       *
*Project:       DMC Mathematical Library                       *
*Originator:    Pascal DORSTER (Texas Intruments)              *
*                                                             *
*Description:   Very simple C main which call Park and         *
*               inverse Park function with cos/sin            *
*               calculation in inverse Park                   *
*                                                             *
*Processor:     C2xx                                           *
*                                                             *
*Status:                                                       *
*                                                             *
*Last Update:   19 Oct 96                                      *
*_____________________________________________________________*
*Date of Mod    |          DESCRIPTION                         *
*---------------|---------------------------------------------*
*               |                                             *
*               |                                             *
*************************************************************/

void PARK(int angle,int ib,int ia,int *iq,int *id);
void INV_PARK(int angle,int iq,int id,int *ia,int *ib,int *ic);

void main()
  {
  int ia,ib,ic,id,iq,angle;
  int cos,sin;


  angle=1000;
  ia=500;
  ib=400;
  PARK(angle,ib,ia,&id,&iq);
  .
  .
  .
  INV_PARK(angle,iq,id,&ia,&ib,&ic);

  }
```

5.8 Clarke_Park function fully C compatible without cos/sin parameters return

```
*************************************************************
*Routine Name:  _PARK                                       *
*Project:       DMC Mathematical Library                    *
*Originator:    Pascal DORSTER (Texas Intruments)           *
*                                                          *
*Description:   Clark + Park calculation                    *
*               without COS & SIN saving for inverse        *
*                calculation C calling funtion, variables  *
*                in C stack.                               *
*                                                          *
*                                                          *
* Calculation:                                             *
*               Ialpha=Ia                                  *
*                Ibeta =1/SQRT(3)*Ia + 2/SQRT(3)*Ib        *
*                with Ia+Ib+Ic=0                           *
*                                                          *
*                isd =Ialpha*COS(angle) + Ibeta*SIN(angle) *
*                isq =-Ialpha*SIN(angle) + Ibeta*COS(angle)*
*                                                          *
*Status:                                                   *
*                                                          *
*Processor:     C2xx                                       *
*                                                          *
*Calling convention:                                       *
*        Input: &Isq, &Isd in stack 16-bits unsigned value *
*               Ia, Ib in stack value 16-bits signed       *
*               Angle in stack value 0-360 degres <=>0h-FFFFh*
*        Output:&Isq, &Isd return parameters via pointers  *
*               Isd, Isq, Sin(angle), Cos(angle) in stack  *
*        Pointed register AR1                              *
*        Fully C calling compatibility                     *
*        Compatible with C interrupt using the stack       *
*                                                          *
*Stack commentary:                                         *
*        Position of current register (AR1) in Caps        *
*        Stack at beginning                                *
*               &isq/&isd/ia/ib/angle/X                    *
*        Stack at return                                   *
*               &isq/&isd/isd/isq/sin/COS                  *
*        Position of current register in Caps              *
*                                                          *
*Function called:                                          *
*        COS_SIN function, cf "SINE, COSINE                *
         on the C2xx" application note                     *
*                                                          *
*Nota:   possibility to delete lines with ;*; commentary   *
*        if the C program doesn't use C function interrupts*
*        and doesn't use the stack in the interrupts routine *
```

```
*        In this case replace all AR2 by AR1 in the function *
*        and add the ADRK instruction in ;*; commentary      *
*                                                            *
*Last Update:   16 Oct 96                                    *
*____________________________________________________________*
*Date of Mod    |          DESCRIPTION                       *
*---------------|--------------------------------------------*
*               |                                            *
*               |                                            *
**************************************************************


  .global       COS_SIN
  .global       _PARK

ONE_BY_SQRT3    .set        1182
TWO_BY_SQRT3    .set        2364

_PARK
                          ;*;Stack context save in case of C
                          interrupt
                          ;*;fonction
  ADRK    4               ;*;reserve memory for temporary storage
  POPD    *+              ;*;pop return address
  SAR     AR1,*           ;*;push AR1
  LAR     AR2,*,AR2
                          ;*;AR2 is temporary register in
                          function
  SBRK    5               ;*
                          ;*;end of stack context save

  SPM     0h
                          ;*STACK : ia/ib/angle/X
  CALL    _COS_SIN        ;calculate SIN & COS of angle
                          ;for vector rotation
                          ;*STACK : ia/ib/sin/COS
  SBRK    3
                          ;*STACK : IA/ib/sin/cos
                          ;calculation ialpha & ibeta with ia, ib
  LT      *+
                          ;*STACK : ia/IB/sin/cos
  MPYK    ONE_BY_SQRT3
                          ;1/sqrt(3)*ia, one_by_sqrt3 in Q11
  LTP     *
  MPYK    TWO_BY_SQRT3
                          ;2/sqrt(3)*ib, two_by_sqrt3 in Q11
  APAC                    ;1/sqrt(3)*ia+2/sqrt(3)*ib
  ADD     #1,10           ;rounding
  SETC    ovm             ;saturation
  RPTK    3
  NORM    *               ;shift one left
  SACH    *               ;
  ADDH    *               ;saturation validation
```

```
                        ;shift one left with overflow
  SACH   *              ;save ibeta
                        ;ialpha = ia
                        ;*STACK : ialpha/IBETA/sin/cos
  LT     *+
  MPY    *+             ;ibeta*sin(angle)
                        ;*STACK : ialpha/ibeta/sin/COS
  LTP    *
  SBRK   3
                        ;*STACK : IALPHA/ibeta/sin/cos
  MPY    *              ;ialpha*cos(angle)
  ADRK   4
                        ;*STACK : ialpha/ibeta/sin/cos/X
  APAC                  ;isd=ibeta*sin+ialpha*cos
  ADD    #1,14          ;rounding
  SACH   *
  ADDH   *              ;overflow management
  SACH   *-
                        ;*STACK : ialpha/ibeta/sin/COS/isd
  LT     *
  SBRK   2
                        ;*STACK : ialpha/IBETA/sin/cos/isd
  MPY    *+             ;ibeta*cos
                        ;*STACK : ialpha/ibeta/SIN/cos/isd
  LTP    *
  SBRK   2
  MPY    *+             ;ialpha*sin
                        ;*STACK : ialpha/IBETA/sin/cos/isd
  SPAC                  ;isq=ibeta*cos-ialpha*sin
  ADD    #1,14
  SACH   *
  ADDH   *
  SACH   *+
                        ;*STACK : ialpha/isq/SIN/cos/isd
  ADRK   2
                        ;*STACK : ialpha/isq/sin/cos/ISD
  LAC    *
  SBRK   4
                        ;*STACK : IALPHA/isq/sin/cos/isd
  SACL   *-             ;C compatible
                        ;***C compatibility parameters****

                        ;*STACK : &isq/&ISD/isd/isq/sin/cos/isd
  LAR    AR5,*-,AR5     ;C compatibity parameters
                        ;*STACK : &ISQ/&isd/isd/isq/sin/cos/isd
  SACL   *,0,AR2        ;C compatibity parameters
  LAR    AR5,*          ;C compatibity parameters
  ADRK   3
                        ;*STACK : &isq/&isd/isd/ISQ/sin/cos/isd
  LAC    *,0,AR5        ;C compatibity parameters
  SACL   *,0,AR1        ;C compatibity parameters
;*;      ADRK   2       ;C compatibity parameters
                        ;*STACK : &isq/&isd/isd/isq/sin/COS/isd
```

```
                        ;*;restore stack context frame
MAR     *-              ;*;
PSHD    *-              ;*;
SBRK    3               ;*;
                        ;*;end restore stack context
RET
```

5.9 Clarke_Park function fully C compatible with cos/sin parameters return

```
*************************************************************
*Routine Name:  _PARK                                       *
*Project:       DMC Mathematical Library                    *
*Originator:    Pascal DORSTER (Texas Intruments)           *
*                                                           *
*Description:   Clark + Park calculation                    *
*               with COS & SIN saving for inverse calculation*
*               C calling funtion, variables in C stack.    *
*                                                           *
* Calculation:                                              *
*               Ialpha=Ia                                   *
*                Ibeta =1/SQRT(3)*Ia + 2/SQRT(3)*Ib         *
*                with Ia+Ib+Ic=0                            *
*                                                           *
*                isd =Ialpha*COS(angle) + Ibeta*SIN(angle)  *
*                isq =-Ialpha*SIN(angle) + Ibeta*COS(angle) *
*                                                           *
*Status:                                                    *
*                                                           *
*Processor:     C2xx                                        *
*                                                           *
*Calling convention:                                        *
*        Input:  &sin, &cos in stack 16-bits unsigned value *
*               &Isq, &Isd in stack 16-bits unsigned value  *
*               Ia, Ib in stack value 16-bits signed        *
*               Angle in stack value 0-360 degres <=>0h-FFFFh*
*        Output:&sin, &cos return parameters via pointers   *
*               &Isq, &Isd return parameters via pointers   *
*               Isd, Isq, Sin(angle), Cos(angle) in stack   *
*        Pointed register AR1                               *
*        Fully C calling compatibility                      *
*        Compatible with C interrupt using the stack        *
*                                                           *
* Stack commentary:                                         *
*        Position of current register (AR1) in Caps         *
*        Stack at beginning                                 *
*               &sin/&cos/&isq/&isd/ia/ib/angle/X           *
*        Stack at return                                    *
*               &sin/&cos/&isq/&isd/isd/isq/sin/COS         *
*                                                           *
*Function called:                                          *
*        COS_SIN function, cf "SINE, COSINE on the C2xx"    *
*        application note                                   *
*                                                           *
*                                                           *
* Nota:  possibility to delete lines with ;*; commentary    *
*        if the C program doesn't use C function interrupts *
*        and doesn't use the stack in the interrupts routine *
```

```
*        In this case replace all AR2 by AR1 in the function *
*                                                             *
*Last Update:   16 Oct 96                                     *
*_____________________________________________________________*
*Date of Mod    |          DESCRIPTION                        *
*---------------|---------------------------------------------*
*               |                                             *
*               |                                             *
***************************************************************

  .global       _COS_SIN
  .global       _PARK

ONE_BY_SQRT3    .set        1182
TWO_BY_SQRT3    .set        2364

_PARK
                            ;*;Stack context save in case of C
                            interrupt
                            ;*;fonction
  ADRK    4                 ;*;reserve memory for temporary storage
  POPD    *+                ;*;pop return address
  SAR     AR1,*             ;*;push AR1
  LAR     AR2,*,            ;*;AR2 is temporary register in
                            function
  SBRK    5                 ;*;
                            ;*;end of stack context save

  SPM     0h
                            ;*STACK : ia/ib/angle/X
  CALL    _COS_SIN          ;calculate SIN & COS of angle
                            ;for vector rotation
                            ;*STACK : ia/ib/sin/COS
  SBRK    3
                            ;*STACK : IA/ib/sin/cos
                            ;calculation ialpha & ibeta with ia, ib
  LT      *+
                            ;*STACK : ia/IB/sin/cos
  MPYK    ONE_BY_SQRT3
                            ;1/sqrt(3)*ia, one_by_sqrt3 in Q11
  LTP     *
  MPYK    TWO_BY_SQRT3
                            ;2/sqrt(3)*ib, two_by_sqrt3 in Q11
  APAC                      ;1/sqrt(3)*ia+2/sqrt(3)*ib
  ADD     #1,10             ;rounding
  SETC    ovm               ;saturation
  RPTK    3
  NORM    *                 ;shift one left
  SACH    *                 ;
  ADDH    *                 ;saturation validation
                            ;shift one left with overflow
  SACH    *                 ;save ibeta
                            ;ialpha = ia
```

```
                        ;*STACK : ialpha/IBETA/sin/cos
LT      *+
MPY     *+              ;ibeta*sin(angle)
                        ;*STACK : ialpha/ibeta/sin/COS
LTP     *
SBRK    3
                        ;*STACK : IALPHA/ibeta/sin/cos
MPY     *               ;ialpha*cos(angle)
ADRK    4
                        ;*STACK : ialpha/ibeta/sin/cos/X
APAC                    ;isd=ibeta*sin+ialpha*cos
ADD     #1,14           ;rounding
SACH    *
ADDH    *               ;overflow management
SACH    *-
                        ;*STACK : ialpha/ibeta/sin/COS/isd
LT      *
SBRK    2
                        ;*STACK : ialpha/IBETA/sin/cos/isd
MPY     *+              ;ibeta*cos
                        ;*STACK : ialpha/ibeta/SIN/cos/isd
LTP     *
SBRK    2
MPY     *+              ;ialpha*sin
                        ;*STACK : ialpha/IBETA/sin/cos/isd
SPAC                    ;isq=ibeta*cos-ialpha*sin
ADD     #1,14
SACH    *
ADDH    *
SACH    *+
                        ;*STACK : ialpha/isq/SIN/cos/isd
ADRK    2
                        ;*STACK : ialpha/isq/sin/cos/ISD
LAC     *
SBRK    4
                        ;*STACK : IALPHA/isq/sin/cos/isd
SACL    *-              ;C compatible
                        ;***C compatibility parameters****

                        ;*STACK : &isq/&ISD/isd/isq/sin/cos/isd
LAR     AR5,*-,AR5      ;C compatibity parameters
                        ;*STACK : &ISQ/&isd/isd/isq/sin/cos/isd
SACL    *,0,AR2         ;C compatibity parameters
LAR     AR5,*           ;C compatibity parameters
ADRK    3
                        ;*STACK : &isq/&isd/isd/ISQ/sin/cos/isd
LAC     *,0,AR5         ;C compatibity parameters
SACL    *,0,AR2         ;C compatibity parameters
                        ;*STACK:&sin/&cos/&isq/&isd/isd/ISQ
                        ;/sin/cos/isd
SBRK    5
                        ;*STACK:&SIN/&cos/&isq/&isd/isd/isq/
                        ;sin/cos/isd
```

```
LAR     AR3,*+
                        ;*STACK:&sin/&COS/&isq/&isd/isd/isq/sin/cos/isd
LAR     AR4,*
ADRK    5
                        ;*STACK:&sin/&cos/&isq/&isd/isd/isq/SIN/cos/isd
LAC     *+,0,AR3
                        ;*STACK:&sin/&COS/&isq/&isd/isd/isq/sin/COS/isd
SACL    *,0,AR2
LAC     *,0,AR4
SACL    *,0,AR1

                        ;*;restore stack context frame
MAR     *-              ;*;
PSHD    *-              ;*;
SBRK    3               ;*;
                        ;*;end restore stack context
RET
```

5.10 Inverse Park function fully C compatible with cos/sin calculation

```
*************************************************************
*Routine Name:  _INV_PARK                                   *
*Project:       DMC Mathematical Library                    *
*Originator:    Pascal DORSTER (Texas Intruments)           *
*                                                          *
*Description:   Inverse Clark + Park calculation            *
*               with COS & SIN calculation                  *
*               Fully C calling compatibility               *
*                                                          *
* Calculation:                                              *
*               Ialpha     = Id*COS(angle)-Iq*SIN(angle)    *
*                Ibeta     = Id*SIN(angle)+Iq*COS(angle)    *
*                                                          *
*               Ia = Ialpha                                 *
*               Ib = -1/2*Ialpha+SQRT(3)/2*Ibeta            *
*               Ic = -1/2*Ialpha-SQRT(3)/2*Ibeta            *
*                                                          *
*Status:                                                   *
*                                                          *
*Processor:     C2xx                                       *
*                                                          *
*Calling convention:                                       *
*        Input: &Ic, &Ib, &Ia in stack 16-bits unsigned value*
*               Id, Iq in stack value 16-bits signed       *
*               Angle in stack value 0-360 degres <=>0h-FFFFh*
*        Output:&Ic, &Ib, &Ic return parameters via pointers *
*               Ia, Ib, Ic in stack                        *
*        Pointed register AR1                              *
*        Fully C calling compatibility                     *
*        Compatible with C interrupt using the stack       *
*                                                          *
*Stack commentary:                                         *
*        Position of current register in Caps              *
*         Stack at beginning                               *
*               &ic/&ib/&ia/id/iq/angle/X                  *
*        Stack at return                                   *
*               &ic/&ib/&ia/ia/ib/ic/X                     *
*                                                          *
*Function called:                                          *
*        COS_SIN function, cf "SINE, COSINE on the C2xx"   *
*        application note                                  *
                                                           *
*                                                          *
* Nota:  possibility to delete lines with ;*; commentary    *
*        if the C program doesn't use C function interrupts  *
*        and doesn't use the stack in the interrupts routine *
*        In this case replace all AR2 by AR1 in this function*
*                                                          *
```

```
*Last Update:   16 Oct 96                                              *
*______________________________________________________________________*
*Date of Mod    |          DESCRIPTION                                  *
*---------------|---------------------------------------------------*
*               |                                                       *
*               |                                                       *
************************************************************************



  .global       _INV_PARK

SQRT3_BY_2      .set        0ddbh

_INV_PARK
                          ;*;Stack context save in case of C
                          interrupt
                          ;*;fonction
  ADRK    4               ;*;reserve memory for temporary
                          storage
  POPD    *+              ;*;pop return address
  SAR     AR1,*           ;*;push AR1
  LAR     AR2,*,AR2       ;*;AR2 is temporary register in
                          function
  SBRK    5               ;*;
                          ;*;end of stack context save

  SPM     0h
  SETC    ovm
  SETC    sxm
                          ;*STACK : id/iq/angle/X
  CALL    COS_SIN         ;calculate SIN & COS of angle
                          ;for vector rotation
                          ;*STACK : id/iq/sin/COS
  LT      *               ;calculation ilpha & ibeta with id, iq
  SBRK    3
                          ;*STACK : ID/iq/sin/cos
  MPY     *+              ;id*cos
                          ;*STACK : id/IQ/sin/cos
  LTP     *+
                          ;*STACK : id/iq/SIN/cos
  MPY     *               ;iq*sin
  SPAC                    ;id*cos-iq*sin
  ADRK    2
                          ;*STACK : id/iq/sin/cos/X
  ADD     #1,14
  SACH    *
  ADDH    *
  SACH    *-
                          ;*STACK : id/iq/sin/COS/ialpha
  LT      *
  SBRK    2
                          ;*STACK : id/IQ/sin/cos/ialpha
  MPY     *+              ;iq*cos
```

```
                        ;*STACK : id/iq/SIN/cos/ialpha
LTP     *
SBRK    2
                        ;*STACK : ID/iq/sin/cos/ialpha
MPY     *+
                        ;*STACK : id/IQ/sin/cos/ialpha
APAC                    ;idsin+iqcos
ADD     #1,14
SACH    *
ADDH    *+
                        ;*STACK : id/iq/IBETA/cos/ialpha
SACH    *


                        ;Ia, Ib, Ic calculation
LT      *-
                        ;*STACK : id/IQ/ibeta/cos/ialpha
MPYK    SQRT3_BY_2      ;Q12
PAC                     ;SQRT(3)/2*ibeta
ADD     #1,11
RPTK    2
NORM    *
SACH    *
ADDH    *
SACH    *
ADRK    3
                        ;*STACK : id/cst*ibeta/ibeta/cos/IALPHA
LAC     *
SBRK    4
                        ;*STACK : ID/cst*ibeta/ibeta/cos/ialpha
SACL    *
                        ;*STACK : IA/cst*ibeta/ibeta/cos/ialpha
LAC     *+,15           ;1/2*Ialpha
                        ;*STACK : ia/CST*IBETA/ibeta/cos/ialpha
ADD     *+,16           ;ic=-1/2*Ialpha-sqrt(3)/2*Ibeta
                        ;*STACK : ia/cst*ibeta/IBETA/cos/ialpha
NEG
SACH    *-
                        ;*STACK : ia/CST*IBETA/ic/cos/ialpha
LAC     *-,16
                        ;*STACK : IA/cst*ibeta/ic/cos/ialpha
SUB     *+,15           ;-Ialpha/2
                        ;*STACK : ia/CST*IBETA/ic/cos/ialpha
SACH    *


                        ;***C compatibility***
                        ;*STACK: &ic/&ib/&ia/ia/IB/ic/cos/ialpha
SBRK    4
                        ;*STACK : &IC/&ib/&ia/ia/ib/ic/cos/ialpha
LAR     AR3,*+          ;AR3 pointed to ic
                        ;*STACK : &ic/&IB/&ia/ia/ib/ic/cos/ialpha
LAR     AR4,*+          ;AR4 pointed to ib
                        ;*STACK : &ic/&ib/&IA/ia/ib/ic/cos/ialpha
LAR     AR5,*+,AR2
```

```
                        ;*STACK : &ic/&ib/&ia/IA/ib/ic/cos/ialpha
LAC     *+,0,AR5
                        ;*STACK : &ic/&ib/&ia/ia/IB/ic/cos/ialpha
SACL    *,0,AR2
LAC     *+,0,AR4
                        ;*STACK : &ic/&ib/&ia/ia/ib/IC/cos/ialpha
SACL    *,0,AR2
LAC     *+,0,AR3
                        ;*STACK : &ic/&ib/&ia/ia/ib/ic/COS/ialpha
SACL    *,0,AR1

                        ;*;restore stack context frame
MAR     *-              ;*;
PSHD    *-              ;*;
SBRK    3               ;*;
                        ;*;end restore stack context

RET
```

5.11 Inverse Park function fully C compatible without cos/sin calculation

```
*****************************************************************
*Routine Name:  _INV_PARK                                       *
*Project:       DMC Mathematical Library                        *
*Originator:    Pascal DORSTER (Texas Intruments)               *
*                                                               *
*Description:   Inverse Clark + Park calculation                *
*               without COS & SIN calculation                   *
*               Fully C calling compatibility                   *
*                                                               *
* Calculation:                                                  *
*               Ialpha = Id*COS(angle)-Iq*SIN(angle)            *
*                Ibeta  = Id*SIN(angle)+Iq*COS(angle)           *
*                                                               *
*               Ia = Ialpha                                     *
*               Ib = -1/2*Ialpha+SQRT(3)/2*Ibeta                *
*               Ic = -1/2*Ialpha-SQRT(3)/2*Ibeta                *
*                                                               *
*Status:                                                        *
*                                                               *
*Processor:     C2xx                                            *
*                                                               *
*Calling convention:                                            *
*        Input: &Ic, &Ib, &Ia in stack 16-bits unsigned value   *
*               Id, Iq in stack value 16-bits signed            *
*               SIN(angle), COS(angle) in stack value Q15       *
*        Output:&Ic, &Ib, &Ic return parameters via pointers    *
*               Ia, Ib, Ic, COS in stack                        *
*        Pointed register AR1                                   *
*        C calling compatibility                                *
*        Compatible with C interrupt using the stack            *
*                                                               *
*Stack commentary:                                              *
*        Position of current register in Caps                   *
*        Stack at beginning                                     *
*               &ic/&ib/&ia/id/iq/sin/cos/X                     *
*        Stack at return                                        *
*               &ic/&ib/&ia/ia/ib/ic/cos/X                      *
*                                                               *
* Nota:  possibility to delete lines with ;*; commentary        *
*        if the C program doesn't use C function interrupts     *
*        and doesn't use the stack in the interrupts routine    *
*        In this case replace all AR2 by AR1 in this function   *
*        and delete the commentary ;*; in start of line         *
*                                                               *
*Last Update:   16 Oct 96                                       *
*_______________________________________________________________*
*Date of Mod    |          DESCRIPTION                          *
*---------------|-----------------------------------------------*
```

```
*                                                               *
*                                                               *
*****************************************************************

  .global       _INV_PARK

SQRT3_BY_2      .set       0ddbh

_INV_PARK
                          ;*;Stack context save in case of C
                          interrupt
                          ;*;fonction
  ADRK   1                ;*;reserve memory for temporary storage
  POPD   *+               ;*;pop return address
  SAR    AR1,*            ;*;push AR1
  LAR    AR2,*,AR2        ;*;AR2 is temporary register in function
  SBRK   2                ;*;
                          ;*;end of stack context save

  SPM    0h
  SETC   ovm
  SETC   sxm
                          ;*STACK : id/iq/sin/cos/X
  MAR    *-
                          ;*STACK : id/iq/sin/COS
  LT     *                ;calculation ilpha & ibeta with id, iq
  SBRK   3
                          ;*STACK : ID/iq/sin/cos
  MPY    *+               ;id*cos
                          ;*STACK : id/IQ/sin/cos
  LTP    *+
                          ;*STACK : id/iq/SIN/cos
  MPY    *                ;iq*sin
  SPAC                    ;id*cos-iq*sin
  ADRK   2
                          ;*STACK : id/iq/sin/cos/X
  ADD    #1,14
  SACH   *
  ADDH   *
  SACH   *-
                          ;*STACK : id/iq/sin/COS/ialpha
  LT     *
  SBRK   2
                          ;*STACK : id/IQ/sin/cos/ialpha
  MPY    *+               ;iq*cos
                          ;*STACK : id/iq/SIN/cos/ialpha
  LTP    *
  SBRK   2
                          ;*STACK : ID/iq/sin/cos/ialpha
  MPY    *+
                          ;*STACK : id/IQ/sin/cos/ialpha
  APAC                    ;idsin+iqcos
  ADD    #1,14
```

```
SACH   *
ADDH   *+
                        ;*STACK : id/iq/IBETA/cos/ialpha
SACH   *

                        ;Ia, Ib, Ic calculation
LT     *-
                        ;*STACK : id/IQ/ibeta/cos/ialpha
MPYK   SQRT3_BY_2       ;Q12
PAC                     ;SQRT(3)/2*ibeta
ADD    #1,11
RPTK   2
NORM   *
SACH   *
ADDH   *
SACH   *
ADRK   3
                        ;*STACK : id/cst*ibeta/ibeta/cos/IALPHA
LAC    *
SBRK   4
                        ;*STACK : ID/cst*ibeta/ibeta/cos/ialpha
SACL   *
                        ;*STACK : IA/cst*ibeta/ibeta/cos/ialpha
LAC    *+,15            ;1/2*Ialpha
                        ;*STACK : ia/CST*IBETA/ibeta/cos/ialpha
ADD    *+,16            ;ic=-1/2*Ialpha-sqrt(3)/2*Ibeta
                        ;*STACK : ia/cst*ibeta/IBETA/cos/ialpha
NEG
SACH   *-
                        ;*STACK : ia/CST*IBETA/ic/cos/ialpha
LAC    *-,16
                        ;*STACK : IA/cst*ibeta/ic/cos/ialpha
SUB    *+,15            ;-Ialpha/2
                        ;*STACK : ia/CST*IBETA/ic/cos/ialpha
SACH   *
                        ;***C compatibility***
                        ;*STACK : &ic/&ib/&ia/ia/IB/ic/cos/ialpha
SBRK   4
                        ;*STACK : &IC/&ib/&ia/ia/ib/ic/cos/ialpha
LAR    AR3,*+           ;AR3 pointed to ic
                        ;*STACK : &ic/&IB/&ia/ia/ib/ic/cos/ialpha
LAR    AR4,*+           ;AR4 pointed to ib
                        ;*STACK : &ic/&ib/&IA/ia/ib/ic/cos/ialpha
LAR    AR5,*+,AR2
                        ;*STACK : &ic/&ib/&ia/IA/ib/ic/cos/ialpha
LAC    *+,0,AR5
                        ;*STACK : &ic/&ib/&ia/ia/IB/ic/cos/ialpha
SACL   *,0,AR2
LAC    *+,0,AR4
                        ;*STACK : &ic/&ib/&ia/ia/ib/IC/cos/ialpha
SACL   *,0,AR2
LAC    *+,0,AR3
                        ;*STACK : &ic/&ib/&ia/ia/ib/ic/COS/ialpha
```

```
    SACL *,0,AR1          ;*;
;*; SACL *,0,AR2
;*; MAR  *+

                          ;*;restore stack context frame
    MAR  *-               ;*;
    PSHD *-               ;*;
                          ;*;end restore stack context

    RET
```