TN135: Three-phase NPC inverter

Introduction

This note deals with the implementation of a three-level Neutral Point Clamped (NPC) inverter.

First, the topology and the theoretical aspects are presented, including a short overview of the modulation techniques.

Then, a possible control implementation on the B-Box RCP or B-Board PRO is introduced, for both C/C++ and ACG implementations.

Finally, an example of a grid-tied NPC converter, derived from AN007 i s presented. Simulation results obtained with Simulink and PLECS are shown.

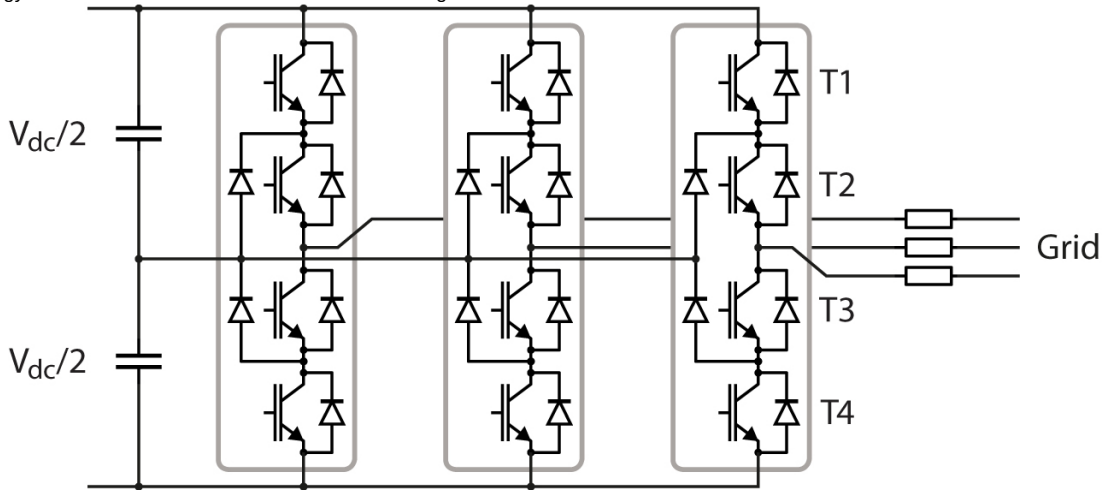
Software resources

File	Modified
File desired_states.m Matlab file containing the enumeration for the desired states: ON, OFF	Jul 02, 2020 by Gabriel Fernandez
File NPC_inverter_CBPWM_PlecsViewer_2015a.slx NPC inverter with CB-PWM, for Simulink	Jul 02, 2020 by Gabriel Fernandez
File NPC_inverter_CBPWM.plecs NPC inverter with CB-PWM, for Plecs	Jul 02, 2020 by Gabriel Fernandez
File NPC_inverter_SVPWM_PlecsViewer_2015a.slx NPC inverter with SV-PWM, for Simulink	Jul 02, 2020 by Gabriel Fernandez
File NPC_inverter_SVPWM.plecs NPC inverter with SV-PWM, for Plecs	Jul 02, 2020 by Gabriel Fernandez
File parameters.m Matlab containing the system parameters	Jul 02, 2020 by Gabriel Fernandez
File states.m Matlab file containing the enumeration for the current state: STANDBY, CHARGING, READY,...	Jul 02, 2020 by Gabriel Fernandez

Download All

Topology

The topology of a three-level NPC inverter is shown in the figure below:



Each leg of the inverter has 4 transistors which can be controlled, giving 2^4=16 total possible states. However, only 3 of these states are feasible since others create short-circuits on the DC-link. The three feasible states give three different output voltages:

- $V_{DC}/2$;
- $-V_{DC}/2$;
- $0\,\mathrm{V}$

The table below shows the conductivity of transistors to get the desired output voltage:

T1	T2	T3	T4	Output voltage	Leg state
ON	ON	OFF	OFF	$V_{DC}/2$	P
OFF	ON	ON	OFF	$0\,\mathrm{V}$	0
OFF	OFF	ON	ON	$-V_{DC}/2$	N

Modulation techniques

Numerous modulation strategies have been proposed for NPC topologies. The selection of the optimum strategy is influenced by numerous factors, such as i) balancing of the neutral point voltage, ii) minimization of the total losses, iii) distribution of losses among the power switches, or iv) AC-side harmonic performance.

This example focuses on the basic implementation of two common techniques, namely carrier-based PWM as well as space vector modulation. More subtle implementations can be considered, which typically also integrate balancing considerations for the DC midpoint voltage. Interesting information can notably be found in [1-2].

Carrier-based PWM

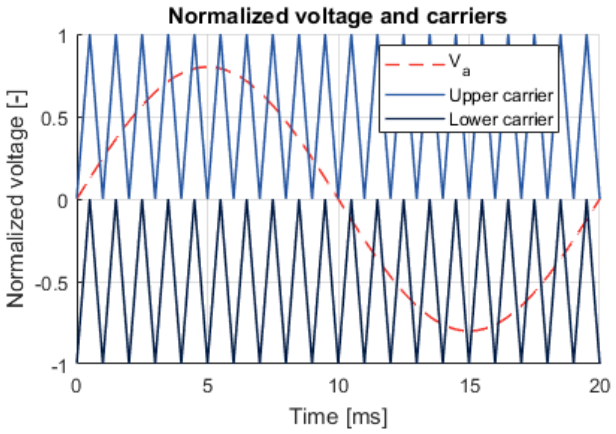
This technique is relatively simple to implement. The principle consists in taking the desired inverter voltages (sometimes also designated as electromotive forces) E^*_{abc}, normalizing and comparing them to two triangular carriers to generate the states of the four transistors of each leg. The normalized voltages have to be in the form:

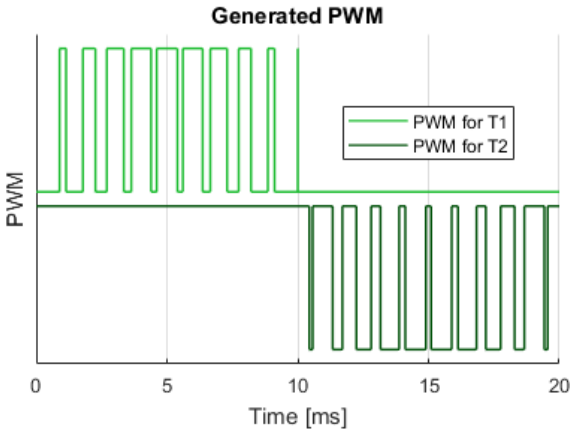
$$E^*_{abc,norm} = m \cdot \sin(\omega t + \phi)$$

With the modulation index m defined as:

$$m = \frac{E^*_{abc,peak}}{V_{dc}/2} \leq 1$$

For each phase, the normalized voltage is compared to the upper carrier, in order to get the state of the transistor T1, and is also compared to the lower carrier to get the state of the transistor T2. The states of the transistors T3 and T4 are the complementary of T1 and T2 respectively.

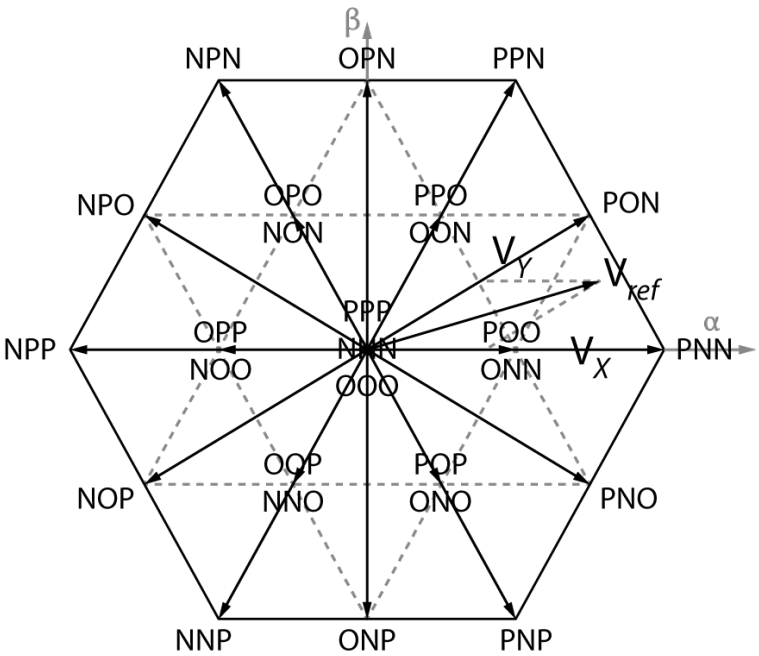




Space vector PWM

A Space Vector Pulse Width Modulation (SV-PWM) is more complex to implement. Each phase has three possible states, which means that the inverter has 27 possible states. Each one of those states can be represented with a space vector in the Clarke referential ():

$$V_{ref} = \frac{2}{3}(V_a + e^{j2\pi/3} \cdot V_b + e^{j4\pi/3} \cdot V_c)$$

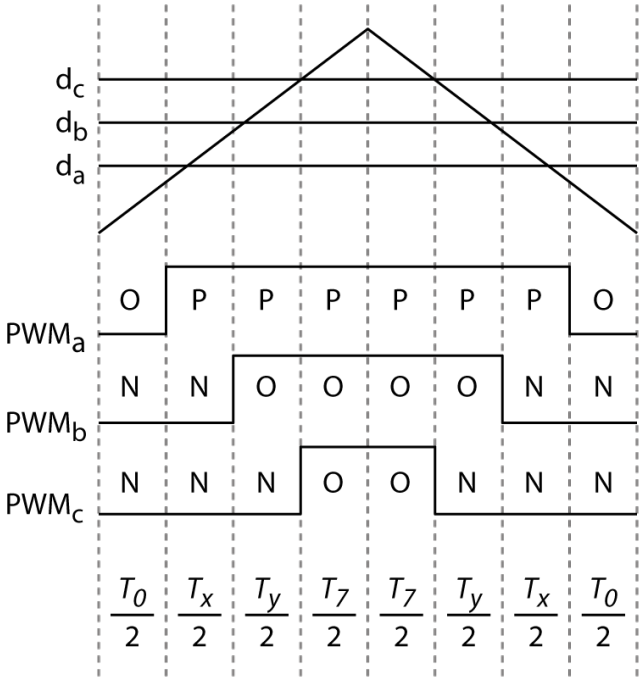


According to the theory of modulation with space vectors, the three vectors V_x, V_y and V_z that are the closest to $V_{\alpha\beta}$ must be found [2]. The dwell time of each vector must be computed according to the formula:

$$V_{ref} \cdot T_{sw} = V_x \cdot T_x + V_y \cdot T_y + V_z \cdot T_z$$

Once the dwell times have been computed for each vector, the switching sequence must be determined. The latter must be selected such that the number of commutations is minimized. For instance, in section n°1, the switching pattern is represented below. This leads to the following duty cycles:

$$d_a = d_x + d_y + d_7 \qquad d_b = d_y + d_7 \qquad d_c = d_7$$



Control and balancing

With respect to the control of AC-side currents and voltages, NPC converters can be controlled similarly to two-level inverters, as most considerations are topology independent. Notably, further information can be found in:

- TN106: Vector current control regarding the current of AC currents in the rotating reference frame.
- TN112: Direct Torque Control for suitable motor drive applications.
- TN110: Proportional Resonant (PR) control regarding AC current control in the stationary frame.

However, NPCs require particular attention to the balancing of the voltage of the DC midpoint, which is not achieved naturally. Indeed, without proper consideration within the chosen modulation algorithm, or suitable closed-loop control, inevitable system unbalances do not tend to compensate, leading excessive and potentially dangerous voltage on one of the two DC busses.

A basic balancing strategy is introduced in TN129: DC bus balancing for NPC inverter.

Academic references

[1] J. Pou, J. Zaragoza, S. Ceballos, M. Saeedifard and D. Boroyevich, "A Carrier-Based PWM Strategy With Zero-Sequence Voltage Injection for a Three-Level Neutral-Point-Clamped Converter," in *IEEE Transactions on Power Electronics*, pp. 642-651, Feb. 2012.

[2] N. Celanovic and D. Boroyevich, "A fast space-vector modulation algorithm for multilevel three-phase converters," in *IEEE Transactions on Industry Applications*, pp. 637-641, March-April 2001.

B-Box / B-Board implementation

C/C++ code implementation

The imperix IDE provides numerous pre-written and pre-optimized functions for both CB-PWM and SV-PWM methods. The necessary parameters are documented within the corresponding .h header file.

As with all peripheral drivers for the B-Box/B-Board, the control implementation is split into two steps:

- The **configuration phase**, executed only once at startup. At this time, six carrier-based modulators (three phases x two pairs of complementary switches) are configured. With space-vector modulation, this is automatically done by the appropriate routines.
- The **run-time phase**, executed regularly at the sampling frequency. At each interrupt call, the duty cycles are updated appropriately. With CB-PWM, this results from the comparison of each voltage (possibly level-shifted) with triangular carriers, while with CV-PWM, the above-described algorithm is used.

CB-PWM implementation

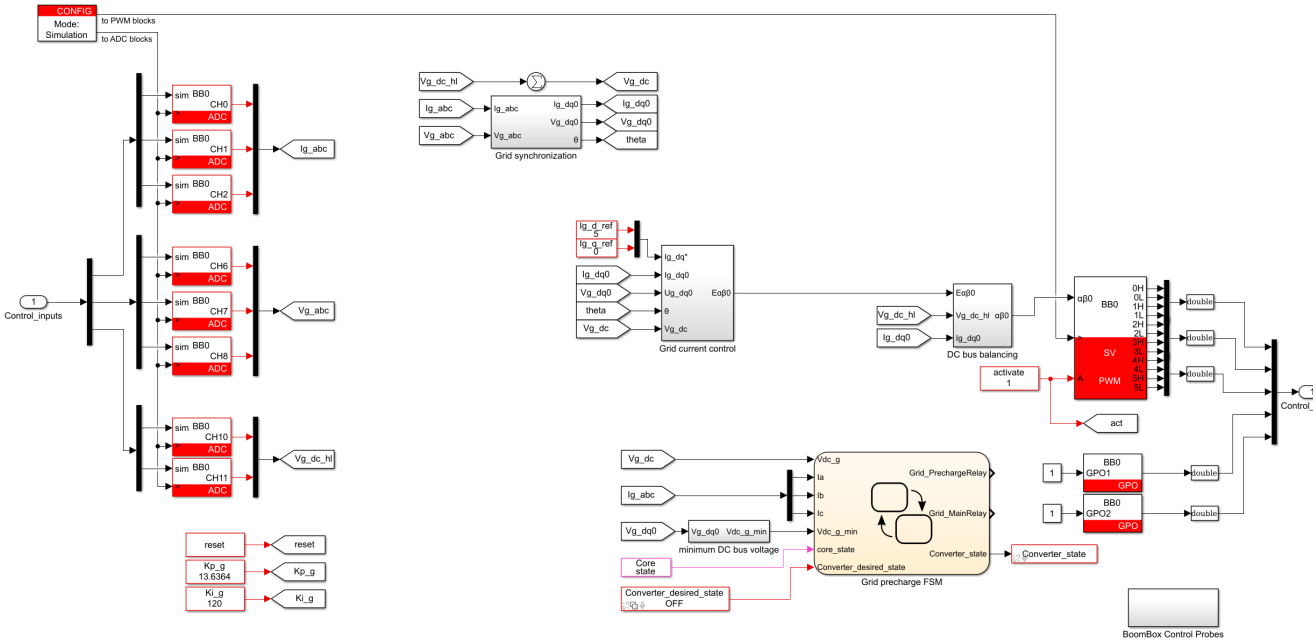
```
tUserSafe UserInit(void)
{
    //... some code

    CbPwm_ConfigureClock(PWM_CHANNEL_0, CLOCK_0);
    CbPwm_ConfigureOutputMode(PWM_CHANNEL_0,
COMPLEMENTARY);
    CbPwm_ConfigureCarrier(PWM_CHANNEL_0,
TRIANGLE);
    CbPwm_ConfigureDeadTime(PWM_CHANNEL_0, 1e-6);
    CbPwm_Activate(PWM_CHANNEL_0);
    // do the same for PWM_CHANNEL_1 to
PWM_CHANNEL_5

        //... some code
        return SAFE;
}

tUserSafe UserInterrupt(void)
{
        //... some code

        //phase A
        CbPwm_SetDutyCycle(PWM_CHANNEL_0, Va);
        CbPwm_SetDutyCycle(PWM_CHANNEL_1, Va+1);
        //phase B
        CbPwm_SetDutyCycle(PWM_CHANNEL_2, Vb);
        CbPwm_SetDutyCycle(PWM_CHANNEL_3, Vb+1);
        //phase C
        CbPwm_SetDutyCycle(PWM_CHANNEL_4, Vc);
        CbPwm_SetDutyCycle(PWM_CHANNEL_5, Vc+1);

        //... some code
        return SAFE;
}
```

SV-PWM implementation

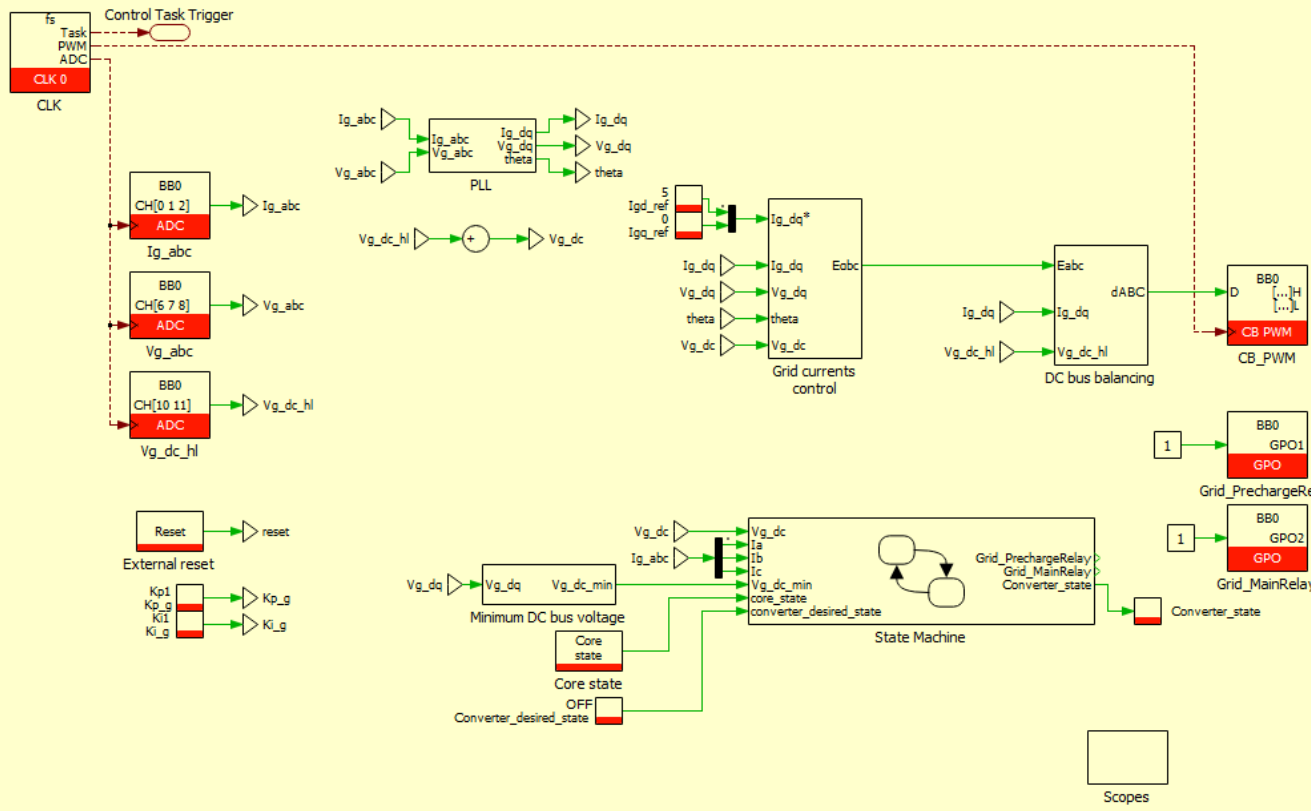
```
tUserSafe UserInit(void)
{
    //... some code

    SvPwm_ConfigureOutputs(SV_MODULATOR_0,
PWM_CHANNEL_0, OPTICAL_PWM_OUTPUT);
    SvPwm_ConfigureLevel(SV_MODULATOR_0, 3);
    SvPwm_ConfigureClock(SV_MODULATOR_0,
CLOCK_0);
    SvPwm_ConfigureOutputMode(SV_MODULATOR_0,
COMPLEMENTARY);
    SvPwm_ConfigureDeadTime(SV_MODULATOR_0, 1e-
6);
    SvPwm_Activate(SV_MODULATOR_0);

    //... some code
    return SAFE;
}

tUserSafe UserInterrupt(void)
{
    //... some code

    SvPwm_RunCartesian(SV_MODULATOR_0, V_aplha,
V_beta, 0);

    //... some code
    return SAFE;
}
```

Simulink and PLECS implementations

As for the graphical implementation, both CB-PWM and SV-PWM implementations can be realized using blocks from the imperix libraries:



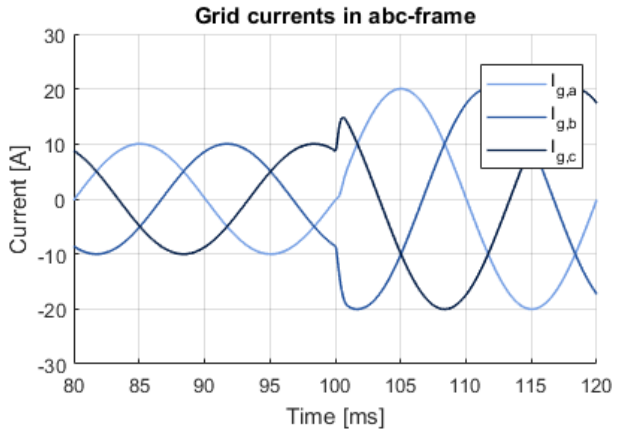
Example of NPC converter control using SVM modulation on Simulink

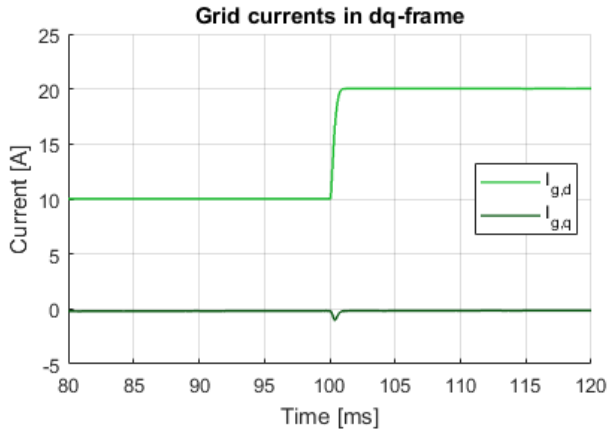


Example of NPC converter control using Carrier-Based modulation on PLECS

Results

The NPC converter was tested with a grid current control on both *d* and *q* axes. A current reference step on the d-axis was performed in simulation mode. The following graph shows the grid currents in *abc*-frame and *dq*-frame:





Going further

An application example using an NPC-type inverter is shown in AN007: Fast eV charger.