IPENCRYPTER.COM

Encrypt HDL Intellectual Property

ipecrypt

compliant with
IEEE Std 1735™-2014 standard

ipencrypter.com

Contents

Encrypt IP

IP author can use *ipecrypt* to encrypt an IP. IP author provides the level of protections through protect directives in common and tool blocks. IP author can choose the tools to support and needs public key for each tool.

Process of creating encrypted IP through *ipecrypt*:

- Create IP in plain text
- Add protect directives around the code to encrypt or alternatively specify directives through a separate file
- Run *ipecrypt* application to generate encrypted IP

***ipecrypt* command syntax for encryption:**

ipecrypt --infile <input> --outfile <output>

Arguments:

--help	produce help message
-I [--infile] arg	input file to encrypt
-O [--outfile] arg	output file encrypted
-F [--force]	force overwrite
-K [--sessionkey] arg	base64 encoded session key for encryption, if empty system generated key will be used
--iv arg	base64 encoded initialization vector for encryption, if empty system generated key will be used
-D [--directive] arg	protect directives, if no directive is found in HDL, the directives from this file will be used
-V [--vhdl]	whether its VHDL file (default is verilog)
--verbose arg	verbosity level [0-4] (fatal, error, warn, info or debug)

Create IP in plain text

Create an IP. Here is a simple counter implemented in Verilog (counter_p.v).

```
//************************************
// Eight-bit Counter
// ipencrypter.com
//************************************
module counter (out,clk,enable,reset);
output [7:0] out;
input        clk, enable, reset;
reg [7:0]    out;

always @( posedge clk )
  begin
    if ( reset )
      out <= 8'b0 ;
    else if ( enable )
      out <= out + 1;
  end
```

```
endmodule
```

Add protect directives

The *protect* directives are added to the IP for encryption. The updated Verilog code is (counter_2e.v) below. In this example “counter” license is required to decrypt IP.

```
//************************************
// Eight-bit Counter
// ipencrypter.com
//************************************
`pragma protect version=2
`pragma protect author="IP author name", author_info="example IP solution providers, visit for more information."
`pragma protect data_method="aes128-cbc"

`pragma protect begin_commonblock
`pragma protect license_proxyname="RMCRYPT"
`pragma protect license_symmetric_key_method="aes128"
`pragma protect license_keyowner="rmcrypt"
`pragma protect license_keyname="rmcrypt_key1"
`pragma protect control decryption=license("counter")?"true":"false"
`pragma protect license_public_key_method="rsa"
`pragma protect license_public_key
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEApu9IvI0gwrFZDPWDEeTX
wMRzBGCu70UZ8V5dCgtoH/7GDjg0vjCdGESJl1UPTbvfnaGHQnBXcM8j/t4BC6BW
Ojd4eIaSnZyfaZndPPWnxFCvBhQRrrZWJrGD5YhV7dOXcqhmkiniGMThviUZ92i0
6FWkOlxio4m9cAxSGV7Qqikr8CQVEqp3fnfeDYt5h7X3Hb3zSmXo6ykS3oxMYpAw
8O0izi7/9+qVXtlGGcxfWfpv4Ztyd9gfZA+oE25cZh2p7/iyNd/pkFJ6X6IRBtkI
SA5KjroIdFx+GljjwyPQGuXkfW1tjhyi0Og83eC/O42RdSqtVtAvPNosiweyE5Vv
fwIDAQAB
`pragma protect end_commonblock

`pragma protect begin_toolblock
`pragma protect key_keyowner = "ipe"
`pragma protect key_method = "rsa"
`pragma protect key_keyname = "ipe_key1"
`pragma protect rights_digest_method="sha256"
`pragma protect key_public_key
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAp1l+cPXg5orGOFto0SjU
Nm3F315sX93gHzhn58IZ5TNes6iqvemnCF8mYJ0EKvj+xkCx0y2lukhzIIqfF+Sp
LnzH61pnO+hVvAvAnDBKu4L85XjpoO8VrM7NpOvpJCviaf2yT9M0pp+EHLz3mZah
Fnae27ugjLZKysl8fgcgchxK4kcTyDs1KP9YNyXwpvdsYyGfhhG0fCX9VPo1eThX
TY6YKmM639a0q/7guCRaSEf89rAznctByl/vgyEWJBnmJMBmFt9WwUositpexhf/
LjvxeI9BKknh0W48KaO7XMyNYN+DKAhMEpmD1G6+o9aBw4BnCb5Ny5+WrGgwzK1O
jQIDAQAB
`pragma protect end_toolblock

`pragma protect begin
module counter (out,clk,enable,reset);
output [7:0] out;
input        clk, enable, reset;
reg [7:0]    out;

always @( posedge clk )
  begin
    if ( reset )
      out <= 8'b0 ;
    else if ( enable )
      out <= out + 1;
  end

endmodule
`pragma protect end
```

For VHDL file instead *pragma protect* use *protect* directive.

Run *ipecrypt* to generate encrypted IP

Command: ipecrypt --infile counter_2e.v --outfile counter_e.v
The encrypted IP generated by *ipecrypt* (counter_e.v) follows.

```
//************************************
// Eight-bit Counter
// ipencrypter.com
//************************************
`pragma protect begin_protected
`pragma protect version=2
`pragma protect encrypt_agent="ipecrypt"
`pragma protect encrypt_agent_info="ipencrypter.com version 0.1"
`pragma protect author="IP author name"
`pragma protect author_info="example IP solution providers, visit for more information."
`pragma protect data_method="aes128-cbc"
`pragma protect begin_commonblock
`pragma protect license_proxyname="RMCRYPT"
`pragma protect license_symmetric_key_method="aes128"
`pragma protect license_keyowner="rmcrypt"
`pragma protect license_keyname="rmcrypt_key1"
`pragma protect control decryption=license("counter")?"true":"false"
`pragma protect license_public_key_method="rsa"
`pragma protect license_public_key
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEApu9IvI0gwrFZDPWDEeTX
wMRzBGCu70UZ8V5dCgtoH/7GDjg0vjCdGESJl1UPTbvfnaGHQnBXcM8j/t4BC6BW
Ojd4eIaSnZyfaZndPPWnxFCvBhQRrrZWJrGD5YhV7dOXcqhmkiniGMThviUZ92i0
6FWkOlxio4m9cAxSGV7Qqikr8CQVEqp3fnfeDYt5h7X3Hb3zSmXo6ykS3oxMYpAw
8O0izi7/9+qVXtlGGcxfWfpv4Ztyd9gfZA+oE25cZh2p7/iyNd/pkFJ6X6IRBtkI
SA5KjroIdFx+GljjwyPQGuXkfW1tjhyi0Og83eC/O42RdSqtVtAvPNosiweyE5Vv
fwIDAQAB
`pragma protect end_commonblock
`pragma protect begin_toolblock
`pragma protect key_keyowner="ipe"
`pragma protect key_method="rsa"
`pragma protect key_keyname="ipe_key1"
`pragma protect rights_digest_method="sha256"
`pragma protect key_block
W3Pd2gYmaSCPJ83X8iaPT9zBc5oizBkE0KUjU4tv29shIHyU8NONGhmULHBC3uKS
3G5p0BVvjkgC7RKNPH8Ps2muRCI0RVyLKutZRa2BVd6AQlcpJc/eHde+IDSv/o4i
rIUSoyxM5mxHC75YEceuLUKqkKjWsRMSwFhbMEZzX7Ft8ErZ1wU9I0tqAkp2RzLC
ROEpOp0+YowfCfBOOpsORQYDWNSx78MBPP6VOr8khCky0fxBx4V0y6VBq7LaX+oV
jb8ETZ6ou2vBhaLZw4mkIrqG2vRlOrCdUH0uFPm8zi/auUR4/MsRkhlG1t95MrOb
X6qFbBqvibQ8OrjmYqjoAg==
`pragma protect end_toolblock="sZqBy8h0Tkzv/LvO3ojKezmc8S8TyzMhDiHSZazLLJM="
`pragma protect data_block
7Xen09bSwpnDWEuo0QmkRCvnt4RtuqoWh7cj8ZMU17ho4AnLsopd4WYvujnWy4R2
UE/b3sC88p416sIAtlXV+2VBuQuXo2okgUZh3Pvvhk+/2UQ0OP57VKiy7EngMJwX
rNvEOJOrb5bmLF39FvgwlxZ9Qisjnt7WIoO7NxIhJvytZ+o5AvnGPymh/t0w3fAp
6n++1yyoZ4hpfNS7kKtLQwS74p1UCNHXd8/BM1v19Z0Mu0RC8xRob3ONSiJbey4C
e6VUTD2NkS6MwUV1fUMmSQOFBfGzAXVMo7ucYpyCSTzyd4m9V3Zc41HQrYMMapS6
`pragma protect end_protected
```

Alternate Way to add protect directives

The *protect* directives can also be specified through a separate file. The sample directive file (directives.txt) follows.

```
`pragma protect version=2
`pragma protect author="Author rmcrypt name", author_info="Author rmcrypt info"
`pragma protect data_method="aes128-cbc"

`pragma protect begin_commonblock
`pragma protect license_proxyname="RMCRYPT"
`pragma protect license_symmetric_key_method="aes128"
```

```
`pragma protect license_keyowner="rmcrypt"
`pragma protect license_keyname="rmcrypt_key1"
`pragma protect control decryption=license("counter")?"true":"false"
`pragma protect license_public_key_method="rsa"
`pragma protect license_public_key
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEApu9IvI0gwrFZDPWDEeTX
wMRzBGCu70UZ8V5dCgtoH/7GDjg0vjCdGESJl1UPTbvfnaGHQnBXcM8j/t4BC6BW
Ojd4eIaSnZyfaZndPPWnxFCvBhQRrrZWJrGD5YhV7dOXcqhmkiniGMThviUZ92i0
6FWkOlxio4m9cAxSGV7Qqikr8CQVEqp3fnfeDYt5h7X3Hb3zSmXo6ykS3oxMYpAw
8O0izi7/9+qVXtlGGcxfWfpv4Ztyd9gfZA+oE25cZh2p7/iyNd/pkFJ6X6IRBtkI
SA5KjroIdFx+GljjwyPQGuXkfW1tjhyi0Og83eC/O42RdSqtVtAvPNosiweyE5Vv
fwIDAQAB
`pragma protect end_commonblock

`pragma protect begin_toolblock
`pragma protect key_keyowner = "ipe"
`pragma protect key_method = "rsa"
`pragma protect key_keyname = "ipe_key1"
`pragma protect rights_digest_method="sha256"
`pragma protect key_public_key
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAp1l+cPXg5orGOFto0SjU
Nm3F315sX93gHzhn58IZ5TNes6iqvemnCF8mYJ0EKvj+xkCx0y2lukhzIIqfF+Sp
LnzH61pnO+hVvAvAnDBKu4L85XjpoO8VrM7NpOvpJCviaf2yT9M0pp+EHLz3mZah
Fnae27ugjLZKysl8fgcgchxK4kcTyDs1KP9YNyXwpvdsYyGfhhG0fCX9VPo1eThX
TY6YKmM639a0q/7guCRaSEf89rAznctByl/vgyEWJBnmJMBmFt9WwUositpexhf/
LjvxeI9BKknh0W48KaO7XMyNYN+DKAhMEpmD1G6+o9aBw4BnCb5Ny5+WrGgwzK1O
jQIDAQAB
`pragma protect end_toolblock
```

For VHDL file instead of *pragma protect* use *protect* directive.

Run *ipecrypt* using directives file to generate encrypted IP

Command: ipecrypt --infile counter_p.v --outfile counter_de.v –directive directives.txt
The encrypted IP generated by *ipecrypt* (counter_de.v) follows.

```
`pragma protect begin_protected
`pragma protect version=2
`pragma protect encrypt_agent="ipecrypt"
`pragma protect encrypt_agent_info="ipencrypter.com version 0.1"
`pragma protect author="Author rmcrypt name"
`pragma protect author_info="Author rmcrypt info"
`pragma protect data_method="aes128-cbc"
`pragma protect begin_commonblock
`pragma protect license_proxyname="RMCRYPT"
`pragma protect license_symmetric_key_method="aes128"
`pragma protect license_keyowner="rmcrypt"
`pragma protect license_keyname="rmcrypt_key1"
`pragma protect control decryption=license("counter")?"true":"false"
`pragma protect license_public_key_method="rsa"
`pragma protect license_public_key
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEApu9IvI0gwrFZDPWDEeTX
wMRzBGCu70UZ8V5dCgtoH/7GDjg0vjCdGESJl1UPTbvfnaGHQnBXcM8j/t4BC6BW
Ojd4eIaSnZyfaZndPPWnxFCvBhQRrrZWJrGD5YhV7dOXcqhmkiniGMThviUZ92i0
6FWkOlxio4m9cAxSGV7Qqikr8CQVEqp3fnfeDYt5h7X3Hb3zSmXo6ykS3oxMYpAw
8O0izi7/9+qVXtlGGcxfWfpv4Ztyd9gfZA+oE25cZh2p7/iyNd/pkFJ6X6IRBtkI
SA5KjroIdFx+GljjwyPQGuXkfW1tjhyi0Og83eC/O42RdSqtVtAvPNosiweyE5Vv
fwIDAQAB
`pragma protect end_commonblock
`pragma protect begin_toolblock
`pragma protect key_keyowner="ipe"
`pragma protect key_method="rsa"
`pragma protect key_keyname="ipe_key1"
`pragma protect rights_digest_method="sha256"
`pragma protect key_block
```

```
aXh6clG8ccZeCgPtc1LoYIYHFLQeANmxsnBhT8TY+F5m5oT8HXaWsRprsR4C3KOh
eDWe99EZ1F1EBzKo/8Hs8ax6Cnb8UJwnGhNeLMKD6/0Dulb9pNRNjjoyNaKvGW+R
3InQYAIolPjaMBkf/wvfpt+O3mUzp9AK5SdCaagoev4K9dusMK0I/eetuQUY4Gh0
EjmH+evLhIii0NsKVHNxYm3YmhKKWYs5QVCU+Y/6S/sPGpIBk1iVDCdrMWg1pp8c
Smu/PohVfoJotPUa/XkIoyH8tjayGe0EH3lVcMup8zrNvCMS+ONWVaoJWLOXJGIs
CnwekdtkqMibrjDOScNmvQ==
`pragma protect end_toolblock="8nAet9TwHNUqP/1eMhFAXjmS2FT/pQwpUopo5Tct1f4="
`pragma protect data_block
5Ka0en0ukVfZVd0RHLs71yrMYyvu5DiI159l/712Y0UAzeUiiT7LIV1MlDuHlH2A
5Mn0/06ClOZV5nI43VBIT9gmQmBtxiaxsSqsklexmU/Ilju42RMOxacErdxfqrwB
inpKWkD7uouUip7W8jMX5HKfpMNel5kXvWPdemLKxljb/UgCM7DSMM9Ze3ZkA7Zn
gNLY9+LJWDZ2lJ6hf385j+G7ABgoZ3a3lAjSBNciFZMqSjXLUFkx9CW7edB/yiWC
3OqE6w2Q37RcWK+RweFu3hAQg47SIO7Q2VVDFLwxpTvlwj2PHzYbmgTLgAmHvUqP
pqARwKypRG9+gN2XRQbwP31P6BAKURXHS8rVgHudqmoyRczhfc2oDqTYwUUEMna7
zw2S/RytyJDAO9FiVZPu0ogAl7l1pJwk0CNAa9y6XtB/IuwdHtH0LoVXyowJEKBA
z8+0UL9mHYFdYxyVLULxgg==
`pragma protect end_protected
```