Protected RAM		
Address	**End address**	**Description**
0x51810040	0x5181005F	AES Protected key 0
0x51810060	0x5181007F	AES Protected key 1
0x51810080	0x5181008F	SM4 Protected key 0
0x51810090	0x5181009F	SM4 Protected key 1
0x518100A0	0x518100AF	SM4 Protected key 2
0x518100B0	0x518100BF	SM4 Protected key 3

7.8.1.3 Countermeasures

CRACEN contains security countermeasures to prevent malicious usage.

The following engines implement countermeasures:

- AES – Masking against Simple Power Analysis (SPA) and Differential Power Analysis (DPA)
- SM4 – Masking against SPA and DPA
- IKG/PKE – Protection against timing attacks and DPA

If CRACEN IKG/PKE is used maliciously, a TAMPC event will be generated and countermeasures enacted according to the TAMPC configuration. The countermeasures are controlled by TAMPC. The bits in TAMPC that control the countermeasures have lock bits.

7.8.1.4 Isolated Key Generator

The Isolated Key Generator (IKG) is a module that derives symmetric and asymmetric keys from the unique seed and optional personalization string.

After IKG has been enabled, CRACEN performs an IKG health test. The CTRDRBGBUSY field of the IKG.STATUS is cleared when the operation has completed. IKG is started by writing to the IKG.START register. The generated IKG keys are valid as long as CRACEN remains enabled. For details on enabling and disabling CRACEN, see ENABLE on page 140.

The IKG derives the following keys from seed upon request:

- One 256-bit ECC P-256 key
- Two 256-bit AES keys

> **Note:** The IKG generated keys are not directly accessible by CPU but are used by the PKE and AES engines. The IKG generated AES keys are not the same as protected keys in protected RAM, but can be used by the same AES engine.

7.8.1.4.1 Loading seed to IKG

The seed used by the IKG to generate keys must be pushed by the KMU to the SEED register and marked as valid before the keys can be generated.

To create and derive a seed the following sequence of operations are needed.

1. Create device unique seed:
   a. Create 3 x 128 bit random number using CRACEN RNG
   b. Provision random number to KMU slots, e.g. 0, 1, and 2 (128 bits in each slot)
      1. SRC.DEST = CRACEN.SEED[n], where n=0, 4, and 8.