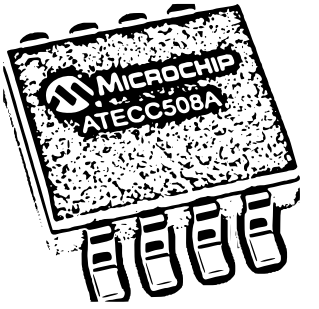
ATECC508A

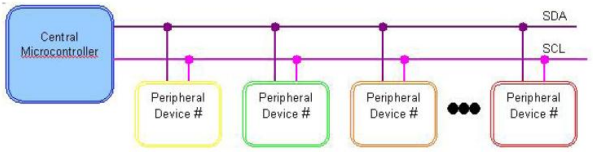
- Easy way to run ECDSA and ECDH Key Agreement
- ECDH key agreement makes encryption/decryption easy
- Ideal for IoT node security
- Authentication without the need for secure storage in the host
- No requirement for high-speed computing in client devices
- Cryptographic accelerator with Secure Hardware-based Key Storage
- Performs High-Speed Public Key (PKI) Algorithms
- NIST Standard P256 Elliptic Curve Support
- SHA-256 Hash Algorithm with HMAC Option
- Host and Client Operations
- 256-bit Key Length
- Storage for up to 16 Keys
- Two high-endurance monotonic counters
- Guaranteed Unique 72-bit Serial Number
- Internal High-quality FIPS Random Number Generator (RNG)
- 10Kb EEPROM Memory for Keys, Certificates, and Data
- Storage for up to 16 Keys
- Multiple Options for Consumption Logging and One Time Write Information
- Intrusion Latch for External Tamper Switch or Power-on Chip Enablement
- Single Wire or I2C Interface
- 2.0V to 5.5V Supply Voltage Range
- 1.8V to 5.5V IO levels
- <150nA Sleep Current
- 8-pad UDFN,8-lead SOIC, and 3-lead CONTACT Packages



Des liens

- http://community.atmel.com/forum/atmel-cryptoauthentication-faqs
  - *I have AES cryptography already, so why do I need anything more? If I am encrypting, isn't that all the security I need?*
  - *I have heard that security has rankings, and that they rank something like this: 1) SHA, 2) AES, 3) RSA, 4) ECC. Does Atmel have the highest ranking?*
  - *Symmetric algorithms like AES and SHA use a single key for all products (e.g. the host and the clients), so if the key is "broken" once, it would then be broken everywhere. Why would I want that?*
  - *How can I justify the cost of an additional device to put security in every system I produce?*
  - *How do I protect the bus between the crypto device and microprocessor?*
  - *How can you encrypt with a one way algorithm like SHA, and isn't XOR weak?*
  - *I have heard that my system isn't secure unless I use a secure micro, is that true?*
  - *How do I protect my valuable software IP?*
- http://www.atmel.com/Images/Atmel-8923S-CryptoAuth-ATECC508A-Datasheet-Summary.pdf

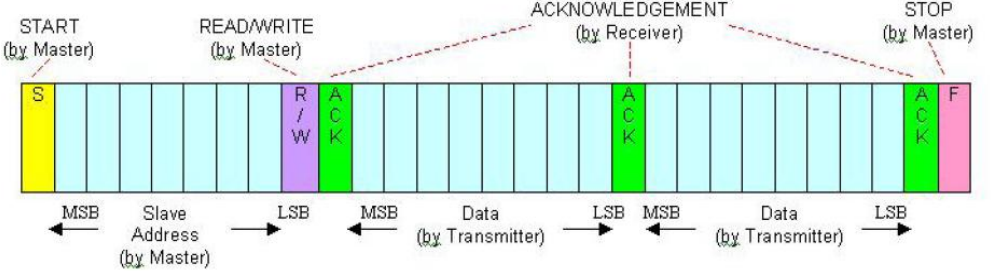
Le bus de communication I2C



Le bus I2C, «*Inter-Integrated Circuit*» :

- un bus générique proposé par Philips dans les années 80, beaucoup utilisé dans les télévisions ;
- synchrone ;
- **débit** : jusqu'à 400 Kbps ;

- seulement 2 signaux :
  - `SCL`, «*Signal Clock*» : le contrôleur «Master» génère l'horloge ;
  - `SDA`, «*Signal Data*» : le «Master» transmets les informations et le «Slave» transmets l'acquittement : si aucun acquittement n'est reçu la communication peut être stoppée ou réinitialisée.



▷ plusieurs «*Slaves*» peuvent être connectés au même bus ;

▷ chaque *Slave* doit disposer d'une **adresse** sur 8bits, composée de :

- une partie fixe qui dépend du constructeur ;
- une partie configurable ;
- le dernier bit qui définit le sens de la communication : 0 pour écrire et 1 pour lire ;
- les communications commencent par un bit de début, «*start bit*», suivi de l'adresse sur 8 bits, le bit d'acquittement, un octet de donnée, un autre bit d'acquittement and à la fin un bit d'arrêt