Ao tentar executar as interrupções desejadas com o GIC-400, percebeu-se que existia alta complexidade em suas ações, o que acarretava em alta latência para suas repostas, algo que não é compatível com o projeto.

Desse modo, percebemos que as interrupções terão de ser feitas pelo Local Interrupt Controler (LIC) do BCM 2836, presente no raspberry 2B que utilizamos.

Quando o LIC recebe uma interrupção, ele a avalia com base nas suas configurações de prioridade e máscara. Se a interrupção estiver habilitada e for de alta prioridade, o LIC envia um sinal ao núcleo do processador. O núcleo, ao receber esse sinal, interrompe a sua execução atual, salva o seu estado e salta para uma rotina de tratamento de interrupção (ISR - Interrupt Service Routine) para lidar com o evento.

Para que tudo isso aconteça com êxito, é necessário endereçar corretamente o que se deseja na memória. O **Local Interrupt Controller** do BCM2836 (fisicamente localizado em 0x40000000 na memória, de acordo com a documentação do kernel Linux) possui um conjunto de registradores mapeados. Desses, os mais relevantes para IPIs são os registradores de **Mailbox**.

Existem 16 mailboxes, com 4 alocados para cada um dos 4 núcleos da Raspberry Pi 2B. O **Mailbox 0** é o mais comum para IPIs em sistemas como o Linux.

O Local timer (0x40000000) conta com comunicação direta com o Local Interrupt Controller, enquanto o ARM timer (0x3F00B400) está em comunicação com o GIC-400, ou seja, para utilizar o LIC para gerar interrupções com base no clock, precisamos utilizar o local timer. Cada núcleo conta com seu próprio local timer, enquanto o ARM timer é único para todo o dispositivo  
  
“**legacy" ARM Timer:**

* **Conexão:** Timer -> GIC -> Núcleo da CPU.
* **Onde vive?** No "Mundo dos Periféricos do SoC" (0x3F00B400).
* **Propósito:** Um timer de uso geral, compatível com o primeiro Raspberry Pi.
* **Fonte:** Documentado na **seção 14 (página 196) do datasheet BCM2835 ARM Peripherals**.

**Local Timer:**

* **Conexão:** Timer -> LIC -> Núcleo da CPU.
* **Onde vive?** Dentro do "Mundo do Complexo ARM", no bloco de periféricos locais de cada núcleo (0x40000000 para o Core 0).
* **Propósito:** Fornecer uma base de tempo de baixíssima latência para cada núcleo individualmente, ideal para o agendador do sistema operacional ou para tarefas de tempo real, como a amostragem de áudio que discutimos.
* **Fonte:** Documentado na arquitetura **ARM Cortex-A7 MPCore** e sua implementação confirmada no device tree bcm2836.dtsi.

